

Arm® Mali™ - IV009 Technical Reference Manual

Version r0p0-00rel0 / Revision r0p0

Revision Information

Date	Issue	Confidentiality	Change
16/03/2018	EAC	Confidential	Initial Release
14/09/2018	REL	Confidential	Initial Release

© Copyright Arm Limited 2018. All rights reserved.

Confidential Proprietary Notice

This document is CONFIDENTIAL and any use by you is subject to the terms of the agreement between you and Arm or the terms of the agreement between you and the party authorised by Arm to disclose this document to you.

This document is protected by copyright and other related rights and the practice or implementation of the information contained in this document may be protected by one or more patents or pending patent applications. No part of this document may be reproduced in any form by any means without the express prior written permission of Arm. No license, express or implied, by estoppel or otherwise to any intellectual property rights is granted by this document unless specifically stated.

Your access to the information in this document is conditional upon your acceptance that you will not use or permit others to use the information: (i) for the purposes of determining whether implementations infringe any third party patents; (ii) for developing technology or products which avoid any of Arm's intellectual property; or (iii) as a reference for modifying existing patents or patent applications or creating any continuation, continuation in part, or extension of existing patents or patent applications; or (iv) for generating data for publication or disclosure to third parties, which compares the performance or functionality of the Arm technology described in this document with any other products created by you or a third party, without obtaining Arm's prior written consent.

THIS DOCUMENT IS PROVIDED "AS IS". ARM PROVIDES NO REPRESENTATIONS AND NO WARRANTIES, EXPRESS, IMPLIED OR STATUTORY, INCLUDING, WITHOUT LIMITATION, THE IMPLIED WARRANTIES OF MERCHANTABILITY, SATISFACTORY QUALITY, NON-INFRINGEMENT OR FITNESS FOR A PARTICULAR PURPOSE WITH RESPECT TO THE DOCUMENT. For the avoidance of doubt, Arm makes no representation with respect to, and has undertaken no analysis to identify or understand the scope and content of, third party patents, copyrights, trade secrets, or other rights.

This document may include technical inaccuracies or typographical errors.

TO THE EXTENT NOT PROHIBITED BY LAW, IN NO EVENT WILL ARM BE LIABLE FOR ANY DAMAGES, INCLUDING WITHOUT LIMITATION ANY DIRECT, INDIRECT, SPECIAL, INCIDENTAL, PUNITIVE, OR CONSEQUENTIAL DAMAGES, HOWEVER CAUSED AND REGARDLESS OF THE THEORY OF LIABILITY, ARISING OUT OF ANY USE OF THIS DOCUMENT, EVEN IF ARM HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

This document consists solely of commercial items. You shall be responsible for ensuring that any use, duplication or disclosure of this document complies fully with any relevant export laws and regulations to assure that this document or any portion thereof is not exported, directly or indirectly, in violation of such export laws. Use of the word “partner” in reference to Arm’s customers is not intended to create or refer to any partnership relationship with any other company. Arm may make changes to this document at any time and without notice.

If any of the provisions contained in these terms conflict with any of the provisions of any signed written agreement covering this document with Arm, then the signed written agreement prevails over and supersedes the conflicting provisions of these terms. This document may be translated into other languages for convenience, and you agree that if there is any conflict between the English version of this document and any translation, the terms of the English version of the Agreement shall prevail.

Words and logos marked with ® or ™ are registered trademarks or trademarks of Arm Limited or its affiliates in the EU and/or elsewhere. All rights reserved. Other brands and names mentioned in this document may be the trademarks of their respective owners. Please follow Arm’s trademark usage guidelines at <http://www.arm.com/about/trademarks/guidelines/index.php>.

Copyright © 2018, Arm Limited or its affiliates. All rights reserved.

Arm Limited. Company 02557590 registered in England.

110 Fulbourn Road, Cambridge, England CB1 9NJ.

LES-PRE-20348

Confidentiality Status

This document is Confidential. This document may only be used and distributed in accordance with the terms of the agreement entered into by Arm and the party that Arm delivered this document to.

Product Status

The information in this document is for a product under development and is not final.

Contents

1	Introduction	13
1.1	Features	13
1.2	Specifications	14
1.3	The design process	18
1.3.1	Implementation	18
1.3.2	Integration.....	18
1.3.3	Programming	18
1.4	Interface and signal description	18
2	Functional description	19
2.1	Principle of operation	19
2.1.1	System interfaces	19
2.2	ISP core architecture	19
2.2.1	Algorithmic dataflow.....	21
2.2.2	Structural dataflow	24
2.2.3	Sensor modes	26
2.3	Programmer's model.....	30
2.4	Configuration and context management.....	33
2.4.1	Conceptual memory map.....	34
2.4.2	Context management modes.....	35
2.5	Functional assumptions	40
2.6	Setting up the ISP.....	41
2.7	Module overview.....	42
2.7.1	Input port	42
2.7.2	Crossbar	47
2.7.3	Video test pattern generator	47
2.7.4	Input formatter	50
2.7.5	Sensor offset WDR and gain WDR	52
2.7.6	Frame stitch.....	52

2.7.7	Decompander0 and Decompander1	57
2.7.8	Digital gain.....	57
2.7.9	Front end sensor offset	58
2.7.10	Square Root	59
2.7.11	Raw frontend	59
2.7.12	Static defect pixel correction	63
2.7.13	Sinter (special noise reduction)	66
2.7.14	Temper (temporal noise reduction)	68
2.7.15	Chromatic Aberration Correction (CAC)	74
2.7.16	Square.....	74
2.7.17	Black Level Correction	74
2.7.18	Lens shading correction.....	75
2.7.19	Static white balance.....	83
2.7.20	Precision management	83
2.7.21	East-west mirror.....	85
2.7.22	Multi-CFA demosaic	85
2.7.23	Purple fringing correction	90
2.7.24	Color correction matrix (4x3).....	90
2.7.25	Color noise reduction (CNR)	92
2.7.26	Non-equidistant gamma and 3D LUT	92
2.7.27	Crop.....	93
2.7.28	Down scaler	94
2.7.29	Sharpen.....	96
2.7.30	RGB gamma.....	96
2.7.31	Color space conversion	97
2.7.32	3A metering statistics.....	99
3	Interrupts.....	109
3.1	Mask vector register.....	111
3.2	Status vector register	111
3.3	Clear vector register.....	111
3.4	Interrupt clear	111
3.5	Shadow structure.....	111

3.6	Level or pulse mode	112
4	Clocking, resets and power management.....	113
4.1	Clock and reset.....	113
4.2	Stalling the video pipeline processing	114
4.3	Low power features	115
4.3.1	Manual module level clock gating	115
5	Error and deadlock considerations	120
6	DMA output format	122
7	Known limitations	124

List of Tables

Table 1.	Configuration parameters	20
Table 2.	CHICKEN_OUT parameters.....	32
Table 3.	WRD stitching	54
Table 4.	Zone statistics	104
Table 5.	Zonal data packing 2 x 32b register.....	105
Table 6.	AF Zone data stored in internal memory.....	105
Table 7.	Clock and clock gate disable conditions	119

Preface

This preface introduces the Arm® Mali™ IV009 Technical Reference Manual. It contains the following sections:

- About this book.
- Feedback.

About this book

This book is for the Arm® Mali-IV009 ISP.

Implementation obligations

This book is designed to help you implement an Arm product. The extent to which the deliverables may be modified or disclosed is governed by the contract between ARM and Licensee. There may be validation requirements, which if applicable will be detailed in the contract between Arm and Licensee and which if present must be complied with prior to the distribution of any silicon devices incorporating the technology described in this document. Reproduction of this document is only permitted in accordance with the licences granted to Licensee.

Arm assumes no liability for your overall system design and performance, the verification procedures defined by Arm are only intended to verify the correct implementation of the technology licensed by Arm, and are not intended to test the functionality or performance of the overall system. You or the Licensee will be responsible for performing any system level tests.

You are responsible for any applications which are used in conjunction with the Arm technology described in this document, and to minimize risks adequate design and operating safeguards should be provided for by you. Arm's publication of any information in this document of information regarding any third party's products or services is not an express or implied approval or endorsement of the use thereof.

Product revision status

The rmpn identifier indicates the revision status of the product described in this book, for example, r1p2, where:

rm Identifies the major revision of the product, for example, r1.

pn Identifies the minor revision or modification status of the product, for example, p2.

Intended audience

This book is for system designers, system integrators, and verification engineers who are designing a System-on-Chip (SoC) device that uses the Mali-IV009 Image Signal Processor (ISP).

Using this book

This book is organized into the following sections:

Section	Description
Introduction	This section introduces you to the Mali-IV009 ISP.
Functional description	This section provides information about the principle of operation, core architecture, context management, and the various modules of the Mali-IV009 ISP.
Interrupts	This section describes the Mali-IV009 ISP interrupt behavior.
Clocking, resets and power management	This section describes the clocking, reset and power management options available with the Mali-IV009 ISP.
Error and deadlock considerations	This section describes the various error and deadlock conditions associated with the Mali-IV009 ISP.
DMA output format	This section provides information about the DMA output format.

Typographic conventions

Font/text type	What it means
<i>italic</i>	Introduces special terminology, denotes cross-references, and citations.
bold	Highlights interface elements, such as menu names. Denotes signal names. Also used for terms in descriptive lists, where appropriate.
monospace	Denotes text that you can enter at the keyboard, such as commands, file and program names, and source code.
<i>monospace italic</i>	Denotes arguments to monospace text where the argument is to be replaced by a specific value.
monospace bold	Denotes language keywords when used outside example code.
<and>	Encloses replaceable terms for assembler syntax where they appear in code or code fragments. For example: MRC p15, 0, <Rd>, <CRn>, <CRm>, <Opcode_2>

Terms and abbreviations

Term/Abbreviation	What it means
ISP	Image Signal Processor
LUT	Look-Up Table
MCU	Micro-Controller Unit
MMU	Memory Management Unit
DOL	Digital Overlay
SoC	System on Chip
HDR	High Dynamic Range
AHB	Advanced High-performance Bus
AXI	Advanced eXtensible Interface
DDR	Double Data Rate
WDR	Wide Dynamic Range
AE	Auto Exposure
AF	Auto Focus
AWB	Auto White Balance
DDP	Dynamic Defect Pixel
HB	Horizontal Blanking
VB	Vertical Blanking
DRC	Dynamic Range Compression

Timing diagrams

This figure explains the components used in timing diagrams. Variations, when they occur, have clear labels. You must not assume any timing information that is not explicit in the diagrams.

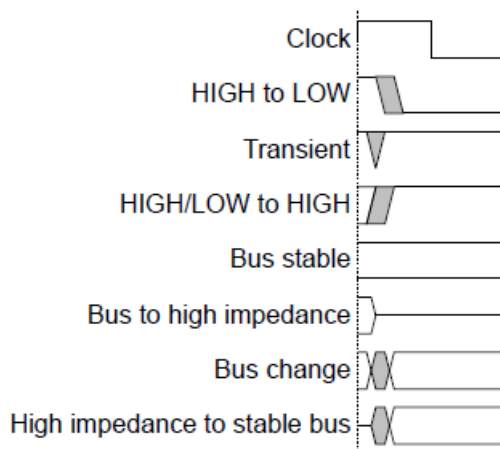


Figure 1. Key to timing diagram conventions

Shaded bus and signal areas are undefined, so the bus or signal can assume any value within the shaded area at that time. The actual level is unimportant and does not affect normal operation.

Signals

The signal conventions are:

Signal level

The level of an asserted signal depends on whether the signal is active-HIGH or active-LOW. Asserted means:

- HIGH for active-HIGH signals.
- LOW for active-LOW signals.

Lower-case n

At the start or end of a signal name denotes an active-LOW signal.

Additional reading

Information published by Arm and by third parties.

See <https://developer.arm.com/> for access to Arm documentation.

Feedback on this product

If you have any comments or suggestions about this product, contact your supplier and give:

- The product name.
- The product revision or version.
- An explanation with as much information as you can provide. Include symptoms and diagnostic procedures if appropriate.

Feedback on content

If you have comments on content then send an e-mail to errata@arm.com. Give:

- The title.
- The number ARM-EPM-136919.
- The page number(s) to which your comments refer.
- A concise explanation of your comments.

Arm also welcomes general suggestions for additions and improvements.

Note: *Arm tests the PDF only in Adobe Acrobat and Acrobat Reader, and cannot guarantee the quality of the represented document when used with any other PDF reader.*

1 Introduction

Mali-IV009 is a multi-camera, multi-exposure high dynamic range (HDR) image signal processor (ISP) for the mid to high end consumer and surveillance market. It brings together some of Arm's most advanced imaging technologies like iridix®, sinter®, temper™ and chromatic aberration correction to provide unrivalled image quality and support to a large number of HDR sensor formats.

This section provides an overview of the ISP and describes its features and modules. All modules may be toggled on or off depending on the needs of the installation.

The following diagram shows how the ISP relates to connected components.

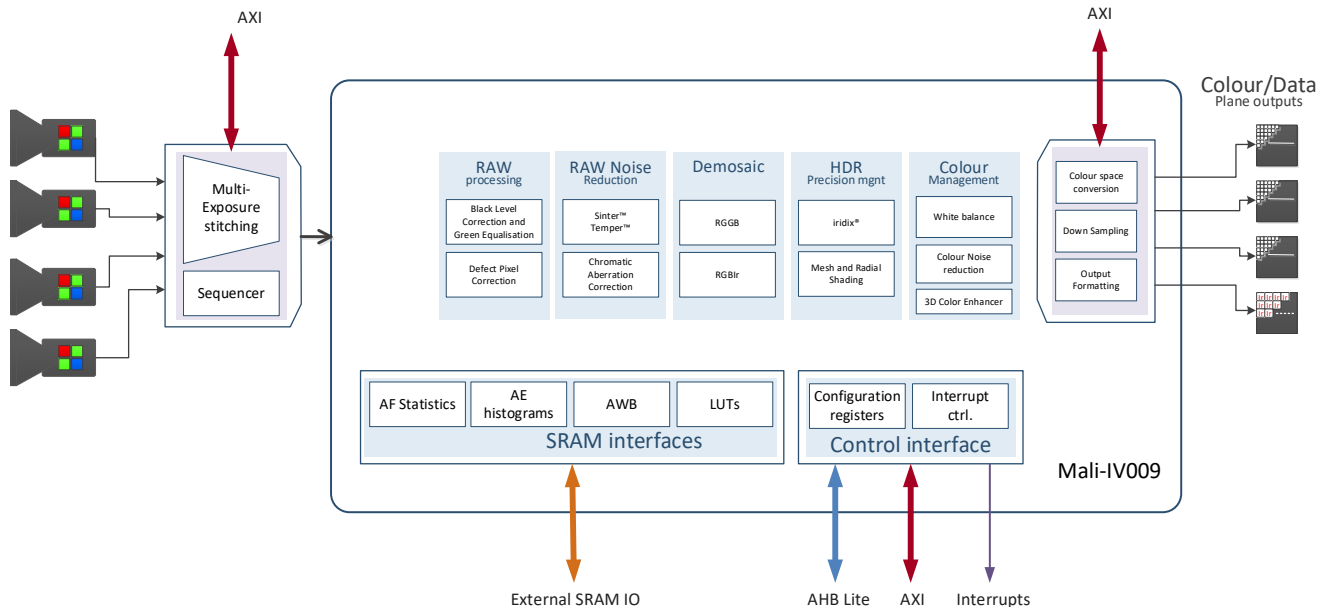


Figure 2. Mali-IV009 functional block diagram

1.1 Features

This section describes the key features of the Mali-IV009

- Multi Camera, Multi Exposure
 - Supports for up to 4 independent cameras. The alignment of different exposures must be done outside the ISP.

- Supports up to 4 exposure HDR (4:1 HDR stitching).
 - Supports up to 16M pixel sensors.
- High Dynamic Range Sensor support
 - Digital overlay (DOL) (up to 4 exposures)
 - On Chip Companded HDR
 - Sensor linearized HDR
- Advanced Imaging
 - Local tone mapping for HDR management using iridix® v8.0
 - Spatial noise reduction using sinter®
 - Temporal noise reduction using temper™
 - Chromatic Aberration and Purple Fringe correction
 - 3D Color look up table
- System Integration and Performance
 - Standard AXI and AHB-lite interfaces
 - Easy DMA based context switching for multi camera support
 - Multiple output formats supported
 - Single pixel per clock architecture

1.2 Specifications

- **Maximum supported resolution**
 - ISP supports up to 16 MPixel resolution per sensor.
 - Customers can configure the ISP to a lower resolution by attaching less delay and limiting the maximum accessible memory address.
 - The following resolutions have been verified in simulations:
 - 16 MP: 4608 x 3456
 - 8 MP: 3840 x 2160, 4096x2160
 - 2 MP: 1920 x 1080

- The minimum resolution verified is 128 x 128.
- ISP should be able to support 16M pixel @30fps on a 16ff process. Target pixel clock approximately 600MHz on 16FF TSMC.

- **Configuration interface**

The MCU can program all the registers and access all memories (except the scratch memories) through the AHB-Lite interface. The interface works in pixel clock. (The clock conversion bridge is the responsibility of customers).

- **AXI interfaces**

- Version AXI3 compliant
- AXI FIFO
 - The FIFO depth is configurable. Customers can configure the depth based on the system latency.
 - Target AXI clock 800 MHz.

- **Video input**

- Input Modes
 - Multi Context (up to 4 single exposure cameras)
 - DOL HDR Support (up to 4 exposure)
 - Onchip Companded
 - Dual Pixel sensors like OV10640
- Pixel format
 - RAW (RGGB)
 - RGBIr (2x2)
- Linear mode input
 - 8, 10, 12, 14, 16, 20 bits
- Native (on sensor) companded input
- Bypass mode video input (for sensors with internal ISP)
 - RGB (12 bits)
- Bypass mode video input (for sensors with internal ISP)

- YUV (10 bits)
- **Raw Domain Processing**
 - HDR sensor decompanding
 - Dynamic Defect Pixel Correction
 - Green channel equalization
 - Static Defect Pixel Correction: 4K locations
 - RAW test pattern generator
 - EW image flip
 - Noise Reduction
 - Spatial Noise Reduction (Sinter)
 - sinter 2.5, 17-line kernel
 - sinter 3, 33-line kernel
 - Temporal Noise Reduction (Temper)
 - Temper 3 width multi-context support
 - Static White balance (Gain and Offsets)
 - Lens shading (vignetting) correction
 - Radial shading correction: 128 nodes
 - Mesh shading correction: 64 x 64
 - Dynamic range compression and local tone mapping
 - Iridix-8 (16x16x32) with Global tone mapping
- **RGB domain processing**
 - Demosaic with anisotropic sharpening control
 - Chromatic Aberration Correction: Supported for Configuration 1 only
 - Color Noise Reduction
 - Purple Fringing Correction
 - Linear matrix color correction

- 3D Color LUT
 - Supported for Configuration 1 only. This is for color enhancement. Feature will provide “favorite colors” options for consumer applications.
- Crop (per o/p channel)
- Gamma correction (per o/p channel)
 - 129 nodes
- scaler (only in the down sampled image output pipe)
 - Supported (16Mpix -> 2Mpix without noticeable quality degradation).
Note: Most use cases will be (8Mpix -> 2Mpix)
- RGB Sharpening (per o/p channel)
- Downscaled output maximum resolution(s)
 - 4096 (4k) (8Mpix)
 - 2k (1080 HD)
- Pixel Variance (Special output for encoder)
- **Video Output**
 - Number of output channels
 - Two – DMA and streaming
 - Color space conversion and Dither (per o/p channel)
 - RGB -> YUV, 4:4:4, 4:2:2, 4:2:0
 - DMA format
 - Non-planar and semi-planar
 - Video output formatter (per o/p channel)
 - As supported by Arm DMA.
See section 6 DMA output format.
- **Metering**
 - AF: 33x33 zones programmable to lower zones by customers
 - AE: 33x33 zones programmable to lower zones by customers

- AWB: 33x33 zones programmable to lower zones by customers
- Histogram: 1024 - Separate weights
- Luminance mean and luminance mean variance stats

1.3 The design process

The Mali-IV009 is delivered as synthesizable RTL. Before it can be used in a product, it must go through the following process:

1.3.1 Implementation

The implementer configures and synthesizes the RTL to produce a hard macrocell.

1.3.2 Integration

The integrator connects the configured design into a SoC. This includes connecting it to a memory system and peripherals.

1.3.3 Programming

This is the last process. System programmers use the following to develop the SoC:

- The software required to configure the Mali-IV009.
- The software required to initialize the Mali-IV009.
- The application software and the SoC tests.

1.4 Interface and signal description

For information related to the interfaces and signal description, refer to the Mali-IV009 Configuration and Integration Guide.

2 Functional description

2.1 Principle of operation

The ISP provides four sensor inputs which may be attributed to different sensors and/or exposures of HDR sensors.

MIPI decoding is not part of the ISP. The ISP video input is a simple data plus strobes parallel-style interface.

In the case of a single sensor, the input is streamed directly to the ISP core. In the case of multiple-exposure HDR, the frames are aligned, prior to combining, by means of AXI connected memory or DDR. This introduces sub-frame latency. In the case of multiple sensors, frames are buffered through AXI connected memory, typically DDR, introducing one frame of latency. Again, the AXI connected memory and interconnect, including AXI controller and MMU, are not part of the ISP deliverable.

The video output of the ISP is available as two streaming outputs and two planes written out through AXI. Each output has configurable access to multiple formats of the image data.

Control of the ISP is through a memory mapped register space, mapped look-up-tables, and statistics memory, over an AHB-lite interface. AHB provides random access into the configuration space. The ISP includes an interrupt controller which provides the necessary synchronization with a processor.

The ISP does not include a processor or micro-controller. However, external processing load is minimized as statistics collection and calculation is implemented within the ISP, and buffer management and configuration context switching is also managed by the ISP hardware.

2.1.1 System interfaces

For information about system interfaces, refer to the Mali-IV009 Configuration and the Integration Guide.

2.2 ISP core architecture

The ISP can be configured as per your requirement using certain configuration parameters.

The following table describes the various configuration parameters:

Parameter name	Description	Supported Values
AXI_WFIFO_AW	Address width of all the DMA writer FIFOs. The FIFO depth is $2^{\text{AXI_WFIFO_AW}}$.	Verified ONLY with 8 and 9.
AXI_RFIFO_AW	Address width of all the DMA reader FIFOs. The FIFO depth is $2^{\text{AXI_RFIFO_AW}}$.	Verified ONLY with 8 and 9.
AXI_WFIFO_UV_AW	Address width of all the UV channel DMA reader FIFOs. The FIFO depth is $2^{\text{AXI_RFIFO_AW}}$.	Verified ONLY with 8 and 9.
DOWN_SCALED_PIPE_CHICKEN_OUT	When this bit is set, the entire downscaled pipe will be removed from design. This includes all the configuration registers for the downscaled pipe. Note: the memory port will still remain and will be internally driven by safe values.	0/1
DMSC_RGBIR_CHICKEN_OUT	When this bit is set, the following conditions prevail: <ul style="list-style-type: none"> • The entire RGBIr demosaic module will be removed. • The Configuration registers will remain inside the design, and the software should not access it. 	0/1
ARM_ISP_LITE_ENABLE	When this bit is set, the following configurations are set inside the ISP: <ul style="list-style-type: none"> • CA correction module will be removed. • Non-equidistant gamma and 3D LUT will be removed. • Sinter version will be set to Sinter 2.5. But <ul style="list-style-type: none"> • Configuration registers will remain inside the ISP. • The configuration address space will be preserved and the software should not access it. • Memory interface signals will still remain. You should not connect any memory. 	0/1

Table 1. Configuration parameters

For more information about configuration parameters and setting, refer to the Mali-IV009 Configuration and Integration Guide.

2.2.1 Algorithmic dataflow

The Mali-IV009 algorithmic architecture is shown in [Figure 4](#). This diagram shows the dataflow through the algorithmic modules, but doesn't show any of the peripheral modules such as the delay line, arbiters, etc. This architecture also assumes that the 4 exposures come aligned at the input interface and there is no data buffer management inside this ISP.

Note: *There are multiplexers between all modules, but they are not shown in the algorithmic block diagram due to space constraints.*

The legend used in the Algorithmic dataflow block diagram is as follows:

- **Green** indicates blocks that can be statically removed.
- A thick Red line indicates primary output.
- The m/n indication between any 2 adjacent modules means:
 - m: is the actual data width of the module.
 - n: when the module is bypassed, n-bits from the input are passed to the output.

For example: In the following figure module A works on 16-bit MSB aligned data (notation m) and the data width of the main pipeline is 20bits MSB aligned (notation n).

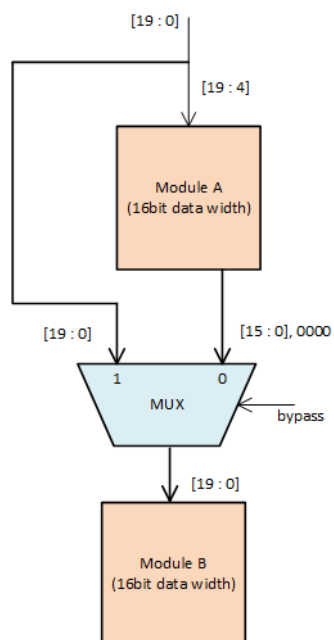


Figure 3. ISP module bypass logic



2.2.2 Structural dataflow

The ISP top level is constructed from multiple clusters which absorb all the modules and peripheral modules it needs. The cluster contains its own configuration, all delay lines, module arbiters, etc. The cluster interface is standard video, clock, memory and bus interface.

The ISP AHB-lite interface is in the vclk clock domain. The Asynchronous AHB-AHB conversion is done outside the ISP. After this conversion the entire ISP configuration space works in the vclk domain.

The ISP configuration register space is split up into cluster level configuration space and each cluster instantiates its own configuration. The address to each cluster is the offset address always starting from 0 (ZERO). There is a common configuration module in the top level which has the entire common configuration for all the clusters.

The ISP configuration interface is standard AHB-lite. The address MUST be either absolute address starting from 0 (ZERO). The ISP has an address decode logic which selects each client based on its address space and further removes the base address to make each client address an offset address.

ISP provides an interrupt handler which provides a single programmable pulse or level interrupt.

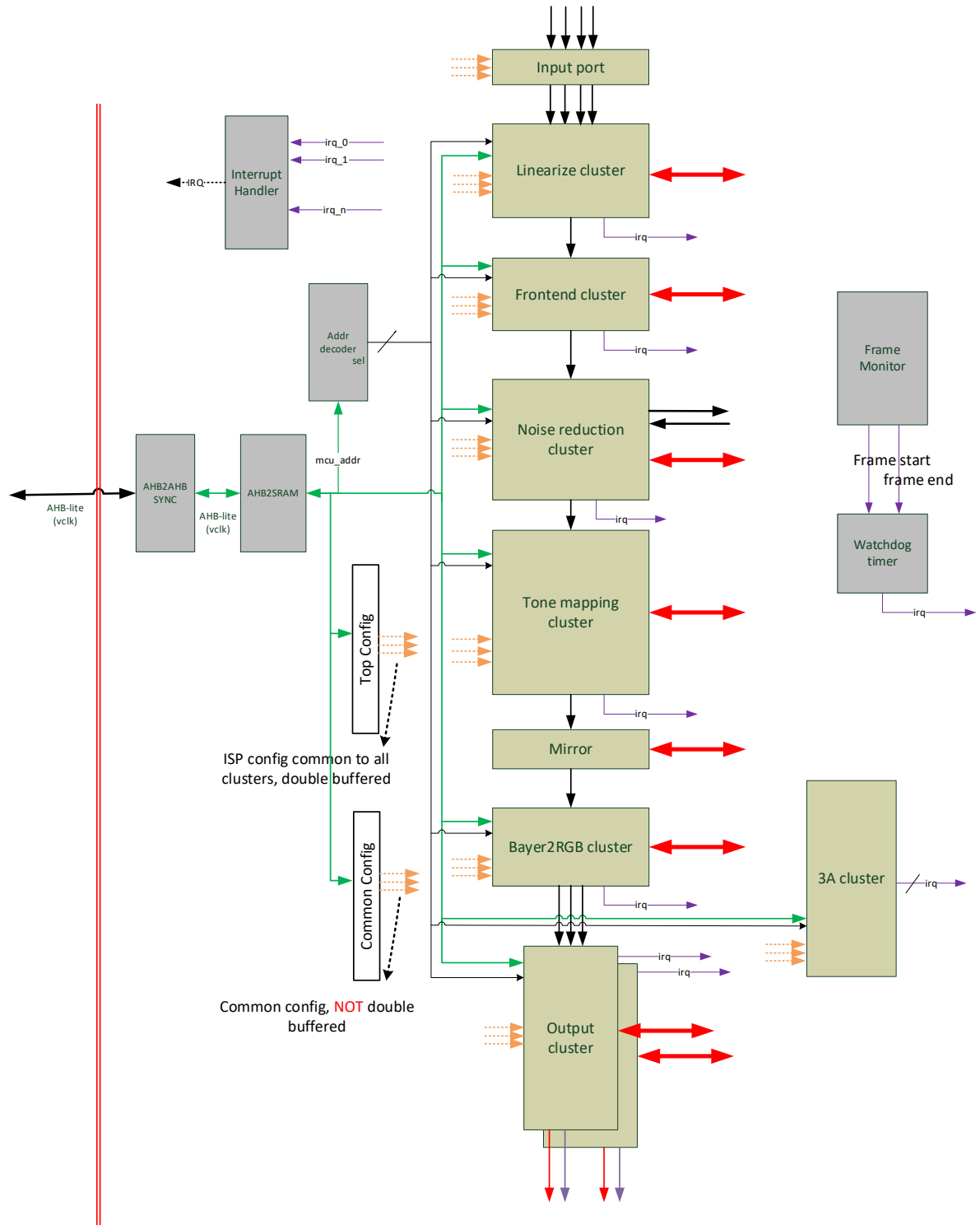


Figure 5. Structural architecture of the ISP

2.2.3 Sensor modes

The ISP supports multiple sensor modes. Some of the special cases are listed in the following table:

Sensor mode	Description
Linear data	Up to 20 bits and MSB aligned.
Sensor companded data	Up to 16 bits and MSB aligned.
3-2 exposures wdr: 16bit Linear + 12bit VS	Supported companded data widths are: 8, 10, 12, 14, 16
3-2 exposures wdr: 12bit companded + 12bit VS	OV2770 or equivalent.
3:2 mux mode	OV10640 3:2 mux mode
16-bit log mode	OV10640 16bit log mode

2.2.3.1 Input data packing

The tables in this section show the input data packing in different sensor modes.

Pixel data is:

- MSB aligned when hvalid_in='1' and vvalid_in='1'
- ignored if hvalid_in='0' or vvalid_in='0'

DOL

vdata0_i → very short exposures

vdata1_i → short exposures

vdata2_i → medium exposures

vdata3_i → long exposures

OV10640 3:2 mux mode

	[19]	[18]	[17:8]	[7:0]
vdata0_i	X	select	L/Vs	XXX
vdata1_i	X	X	S	XXX
vdata2_i	X	X	XXX	XXX
vdata3_i	X	X	XXX	XXX

OV2770 16-bit linear + 12 bit VS mode

	[19:8]	[7:4]	[3:0]
vdata0_i	16-bit linear		X
vdata1_i	BS	X	X
vdata2_i	X	X	X
vdata3_i	X	X	X

OV2770 12-bit companded + 12 bit VS mode

	[19:8]	[7:0]
vdata0_i	12-bit companded	X
vdata1_i	VS	X
vdata2_i	X	X
vdata3_i	X	X

All other modes

	[19:n]	[n-1:0]
vdata0_i	MSB aligned	X
vdata1_i	X	X
vdata2_i	X	X
vdata3_i	X	X

RGB/YUV bypass

	[19:n]	[n-1:0]
vdata0_i	Y/R (MSB aligned)	X
vdata1_i	U/G (MSB aligned)	X
vdata2_i	V/B (MSB aligned)	X
vdata3_i	X	X

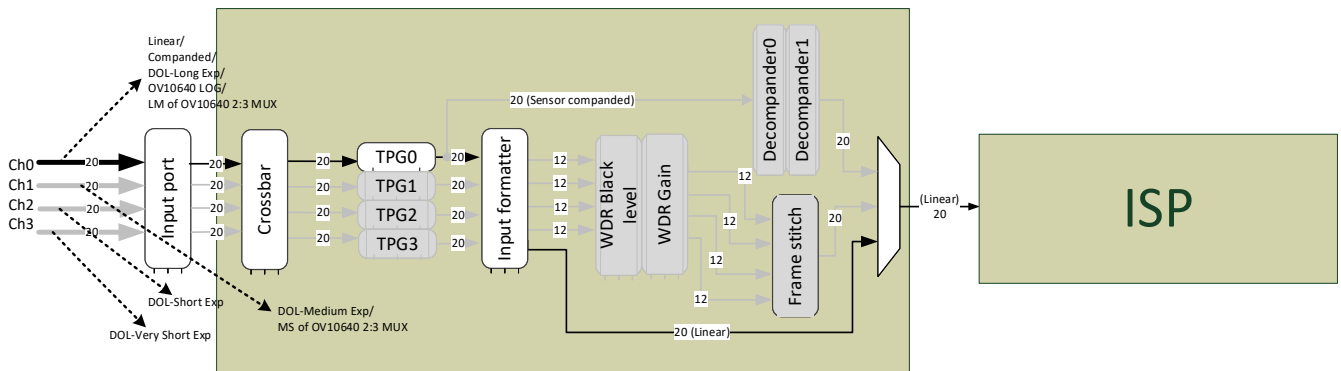
2.2.3.2 Linear or sensor companded where slopes are 2^n

In this mode the input data must be sent on channel-0 and must be MSB aligned. The crossbar must be in the default configuration so that the input data is directly mapped to its same output channel. The input formatter must be configured with “linear” input mode and correct bit-width must be selected. For example, if the bit-width selected is 14bits, then the input port takes only 14 MSB from the input data and appends 0 for the remaining bits.

The `linear data src` register must select sensor linear data mode.

Please refer to the **ISP_CONFIG_PING: top** and **ISP_CONFIG_PONG: top** sections in the Mali-IV009 register map for the detailed register descriptions.

In the following figure the white blocks indicate the active blocks and dark black lines indicate the actual signal flow. The grey blocks and signal lines are inactive.



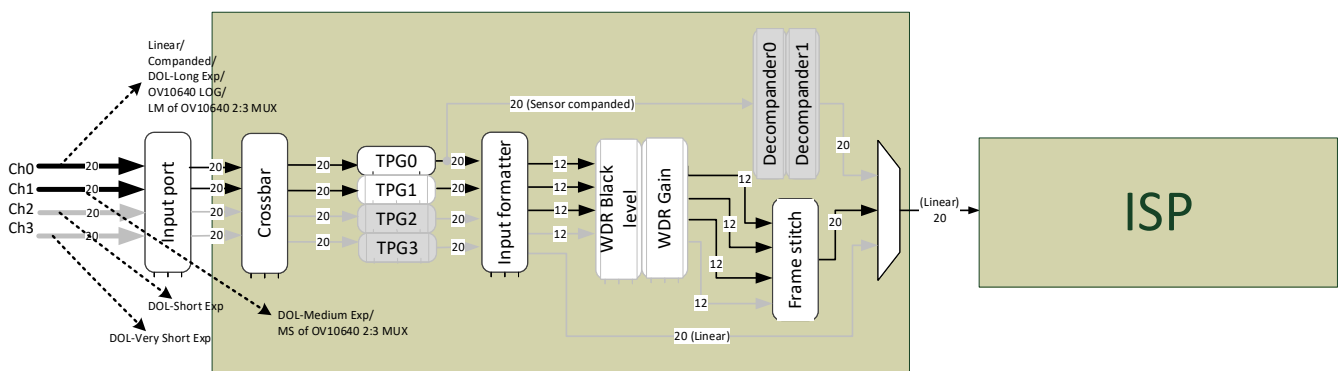
2.2.3.3 Ov10640 2-3 MUX

In this mode 2 exposures are sent on channel-0 and channel-1 as per the tables described in the section **Input data packing**. The crossbar must be in the default configuration so that the input data is directly mapped to its same output channel. The input formatter must be configured with the correct input mode.

The input formatter converts 2 exposures to 3 exposures and the resultant 3 exposures are mapped to 3 outputs of the input formatter. Frame stitch is programmed in 3-exposure stitching mode and the `linear data src` register is set in the frame stitch data source mode.

Please refer to the **ISP_CONFIG_PING: top** and **ISP_CONFIG_PONG: top** sections in the Mali-IV009 register map for the detailed register descriptions.

In the following figure the white blocks indicate the active blocks and dark black lines indicate the actual signal flow. The grey blocks and signal lines are inactive.

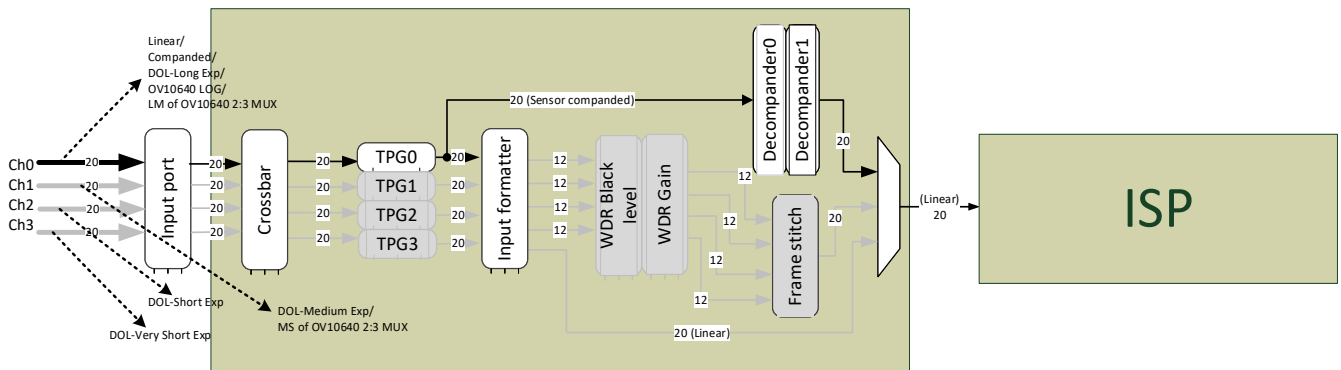


2.2.3.4 Log or sensor companded data where slopes are NOT 2^n

In this mode the input data must be sent on channel-0 and must be MSB aligned. The crossbar must be in the default configuration so that the input data is directly mapped to its same output channel. The input formatter is bypassed. The decompander-0 (33 nodes) and decompander-1 (257 nodes) are programmed to create an equivalent log curve. The `linear data src` register is programmed appropriately to select the decompanded data.

Please refer to the **ISP_CONFIG_PING: top** and **ISP_CONFIG_PONG: top** sections in the Mali-IV009 register map for the detailed register descriptions.

In the following figure the white blocks indicate the active blocks and dark black lines indicate the actual signal flow. The grey blocks and signal lines are inactive.



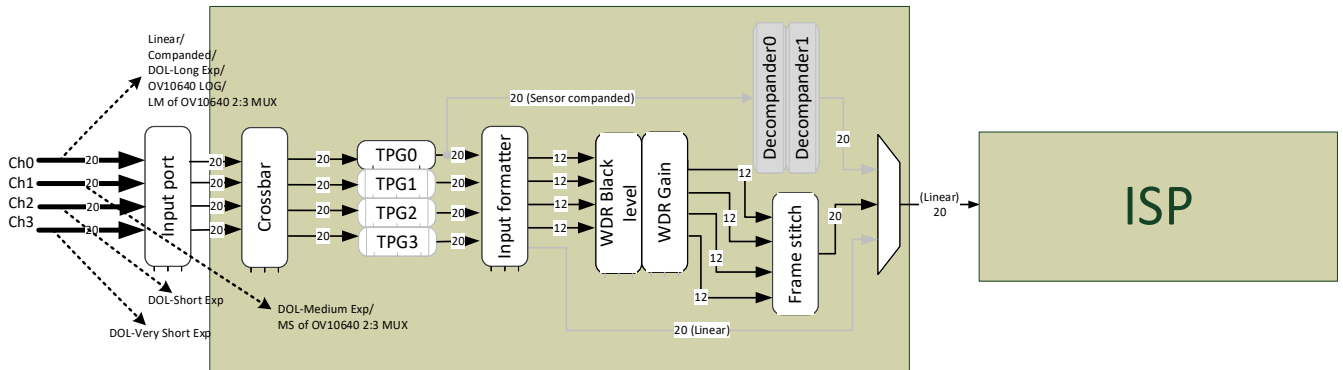
2.2.3.5 Four-exposure DOL

In this mode the 4 exposures are mapped to 4 input channels as per the tables described in the section Input data packing. The crossbar must be in the default configuration so that the input data is directly mapped to its same output channel. The input formatter can either be bypassed or used in pass-through mode by selecting the correct bit-width.

Frame stitch is programmed in 4-exposure stitching mode and the `linear data src` register is set in the frame stitch data source mode.

Please refer to the **ISP_CONFIG_PING: top** and **ISP_CONFIG_PONG: top** sections in the Mali-IV009 register map for the detailed register descriptions.

In the following figure the white blocks indicate the active blocks and dark black lines indicate the actual signal flow. The grey blocks and signal lines are inactive.



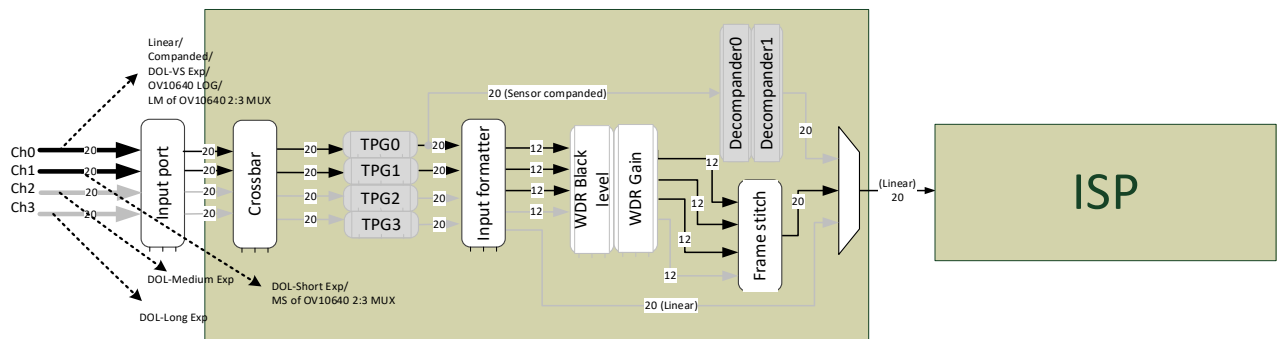
2.2.3.6 OV2770 3:2 exposure support

In this mode the 16bit linear or the 12 bit companded data comes on channel-0 MSB aligned and the VS exposure comes on the channel-1 MSB aligned. The input formatter must be configured with the correct input mode.

The input formatter converts 2 exposures to 3 exposures and the resultant 3 exposures are mapped to 3 outputs of the input formatter. Frame stitch is programmed in 3-exposure stitching mode and the linear data src register is set in the frame stitch data source mode.

Please refer to the **ISP_CONFIG_PING: top** and **ISP_CONFIG_PONG: top** sections in the Mali-IV009 register map for the detailed register descriptions.

In the following figure the white blocks indicate the active blocks and dark black lines indicate the actual signal flow. The grey blocks and signal lines are inactive.



2.3 Programmer's model

For information regarding the ISP register programming, refer to the Mali-IV009 Register Map document.

The ISP provides some static parameters through which certain modules can be permanently removed from ISP.

You must read the status of the **ISP_COMMON:isp global parameter status** register to know if a certain module is statically removed or not. This register, shown in [Figure 6](#), indicates the status of that CHICKEN BIT.

Warning *If a module is statically removed, programmers must not access any of the registers and memory (if any).*

The following figure shows the **isp global parameter status** register

Addr	Mode	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	Default
0x00068	R/O	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	--
													sinter version	LUT 3d	ds pipe	cac	dmisc rgbir	

Figure 6. ISP global parameter status register

The following table describes the various CHICKEN_OUT parameters:

R/O register name	Description
dmisc rgbir	0: Demosaic RGBIr is present in the design. 1: Demosaic RGBIr is statically removed from the ISP. The S/W must not access the ISP_CONFIG_PING:demosaic rgbir and the ISP_CONFIG_PONG:demosaic rgbir registers.
cac	0: CA Correction is present in the design. 1: CA Correction is statically removed from the ISP. The S/W must not access the ISP_CONFIG_PING:ca correction and the ISP_CONFIG_PONG:ca correction registers and must also not access the following memories <ul style="list-style-type: none"> • CA_CORRECTION_FILTER_PING_MEM • CA_CORRECTION_FILTER_PONG_MEM • CA_CORRECTION_MESH_PING_MEM • CA_CORRECTION_MESH_PONG_MEM.

R/O register name	Description
ds pipe	<p>0: DS pipeline is present in the design. 1: DS pipeline is statically removed from ISP. The S/W must not access the following register spaces:</p> <ul style="list-style-type: none"> • ISP_CONFIG_PING:ds crop and ISP_CONFIG_PONG:ds crop • ISP_CONFIG_PING:ds scaler and ISP_CONFIG_PONG:ds scaler • ISP_CONFIG_PING:ds gamma rgb and ISP_CONFIG_PONG:ds gamma rgb • ISP_CONFIG_PING:ds sharpen and ISP_CONFIG_PONG:ds sharpen • ISP_CONFIG_PING:ds cs conv and ISP_CONFIG_PONG:ds cs conv • ISP_CONFIG_PING:ds dma writer and ISP_CONFIG_PONG:ds dma writer • ISP_CONFIG_PING:ds uv dma writer and ISP_CONFIG_PONG:ds uv dma writer <p>Also, the S/W must not access the following memories:</p> <ul style="list-style-type: none"> • DS_SCALER_HFILT_COEFMEM • DS_SCALER_VFILT_COEFMEM • DS_GAMMA_RGB_PING_MEM • DS_GAMMA_RGB_PONG_MEM
LUT 3d	<p>0: sRGB gamma and 3D LUT is present in the design. 1: sRGB gamma and 3D LUT are statically removed from ISP. The S/W must not access the following register spaces:</p> <ul style="list-style-type: none"> • ISP_CONFIG_PING: nonequ gamma • ISP_CONFIG_PING: nonequ gamma <p>Also, the S/W must not access the following memories:</p> <ul style="list-style-type: none"> • LUT3D_MEM
sinter version	<p>0: SINTER2.5 is used 1: SINTER3 is used</p>

Table 2. CHICKEN_OUT parameters

Mali-IV009 has a reserved chicken bit register which should be kept in its default values.

Register name: **isp_global_chicken_bit**

Addr: 0x0064

Default: 0x00000004

bit[0] : **Reserved**

bit[1] : **isp_global_chicken_bit_rd_start_sel**

0 = temper dma reader will start reading based on the frame start.

1 = temper dma reader will start reading based on linetick of the dma writer.

bit[2] : **Reserved**

bit[3] : **Reserved**

bit[4] : **isp_global_chicken_bit_soft_rst_apply_immediately**

0 = When this chicken bit is set to 0, the soft reset will be stretched by 1024 cycles so that it is asserted till the pipeline delay of internal acl signals.

1 = When this set to 1, the soft reset will not be stretched internally. The SW must make sure its kept high enough.

bit[5] : **Reserved** .

2.4 Configuration and context management

The ISP consists of two physical address spaces called Ping and Pong. These are named so because they can operate alternately one at a time. The ping and pong address spaces along with memory mapped configuration spaces (see the Mali-IV009 Register Map for more details) provide the ability to handle **multiple contexts**.

A **Context** is defined as the address space of a single sensor. When multiple sensors are connected to the ISP, the context management is referred to as multi-context management.

The Mali-IV009 ISP supports up to 4 sensors thus enabling it to work in the following modes:

- **Single context mode:** a combination of one sensor and 1 memory space.
- **Multi-context mode:** a combination of more than one sensor (maximum of 4) and memory spaces.

The top-level configuration signals and the LUT registers are double buffered through the ping-pong structure. This means that when one frame is written to the ping space the

space is locked and the next frame is buffered in to the pong space to be shown next. All the register space switches from ping and pong globally and the ISP does not work in overlapping video input and output. This means the current output frame and next input frame must not be active at a time.

2.4.1 Conceptual memory map

The following figure shows the conceptual memory map:

Common config space: The common config space is not ping-ponged and is common for all contexts. Any value written to this space comes into effect immediately.

Implicit ping-pong space: This shadow buffer address space is not visible to the MCU. The module exposes only 1 address space but internally switches between ping and pong.

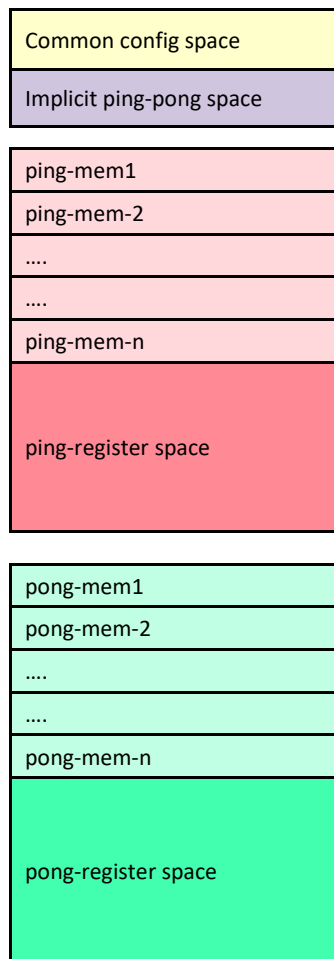


Figure 7. Conceptual memory map

2.4.2 Context management modes

The following modes of handshaking between software (SW) and hardware (HW) are available for configuration and context management:

- Software only mode
- Single context SW-HW mode
- Multi-context SW-HW Mode

2.4.2.1 Software only mode

In this mode the entire context management is handled only through software.

The sequence of event in this mode is as follows:

1. The MCU must write '1' to `mcu_override_config_select` (see the register map for more details) in the common configuration space to take full control of the context management.
2. The MCU decides whether to use PING or PONG address space and program accordingly.
3. The MCU then indicates this to the ISP through the `mcu_ping_pong_config_select` signal.

In the software only mode the MCU can use the ISP in ping only or pong only mode.

In this case, the hardware provides additional protection in the active frame duration as follows:

- When the software changes the status of `mcu_ping_pong_select` during an active frame time, the value takes effect only from the nearest vertical blanking thus protecting the ongoing frame data from corruption.
- In the active frame time, any attempt to update the configuration set is ignored.

2.4.2.2 Single context SW-HW mode

In this mode, the `mcu_override_config_select` must be 0 and the `multi_context_mode` must be `SINGLE_CTX`. The configuration programming is always Ping-Pong alternating. The reset value of the select is always PING.

Note: *In this mode, the SoC system must ensure that the configuration for an address space is ready before the frame is sent for processing.*

Figure 8 shows the flow of the single context SW-HW mode after a reset. The ping pong selection always starts from PING.

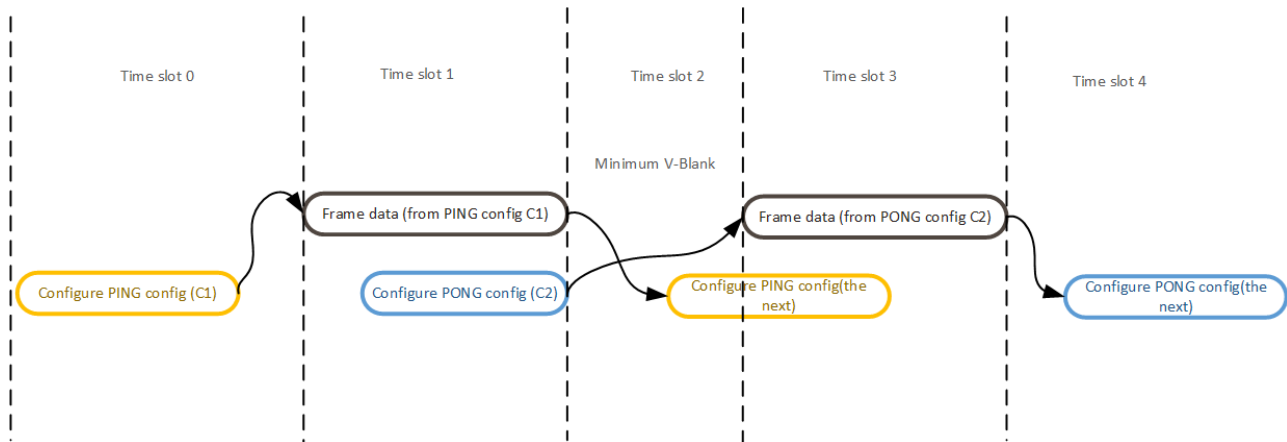


Figure 8. Single context SW-HW mode

Referring to Figure 8, the sequence of events in the single context SW-HW mode is as follows:

1. Time slot 0:
 - a. The MCU initiates the configuration programming in the vertical blanking period.
 - b. The Ping space is configured during this interval and the configuration end ahead of the incoming frame data.
2. Time slot 1:
 - a. The frame data based on the ping configuration is ready to process.
 - b. The ISP locks the Ping space during this interval and it cannot be changed.
 - c. The Pong space is configured during this interval and the configuration end before the next incoming frame.
3. Time slot 2:
 - a. This is the vertical blanking interval and the MCU must maintain it to the minimum.
 - b. The next Ping configuration can start during vertical blanking.
4. Time slot 3:

- a. Frame data from the pong is processed from the beginning of this interval, that is, after the vertical blanking.
 - b. The ISP locks the Pong space during this interval and it cannot be changed.
 - c. While the Pong space data is being processed, the Ping space can be configured simultaneously.
5. Time slot 4:
 - a. The next Ping space data is ready.
 - b. The ISP locks the Ping space during this interval and it cannot be changed.
 - c. The next Pong space is configured during this interval.
6. This sequence continues and keeps on alternating sequentially between Ping and Pong.

Note: *You must ensure that the configuration for a frame is completed before that frame is sent to the ISP.*

2.4.2.3 Multi-context SW-HW Mode

In this mode, the `mcu_override_config_select` must be 0 and the `multi_context_mode` must be `MULTI_CTX`. The configuration programming is always Ping-Pong alternating. The reset value of the select is always PING.

The SoC system must perform the time-division-multiplexing of the frames from different sensors. There is handshaking between the ISP, external buffer management and the MCU for the context management. The feedback from the ISP is optional and you may choose to ignore it.

The ISP provides a signal called `isp_multictx_frame_req` in the primary output port. This signal indicates when the ISP is ready to accept a new frame. This signal is a level signal which goes LOW when the new frame starts coming. This signal goes high when both of the following conditions are true:

- The MCU has configured the PING/PONG address space.
- The ISP is NOT busy processing the current frame.

The ISP configuration address space has 2 registers called **ISP_CONFIG_PING:multi ctx:Config done** and **ISP_CONFIG_PONG:multi ctx:Config done**. These registers must be programmed after the configuration is completed. The hardware detects the rising edge of the `config_done` signal to recognise that the configuration is completed.

NOTE: *These registers are the last registers in their respective address space. If the SoC system is DMA-ing the config, then this register will be written automatically.*

You can decide NOT to use the `isp_multictx_frame_req` and make sure that the Minimum vertical blanking (please refer to Functional assumptions section) is maintained and the configuration is done before the data is sent.

The following diagram shows the steps involved in the configuration of the PING/PONG address spaces.

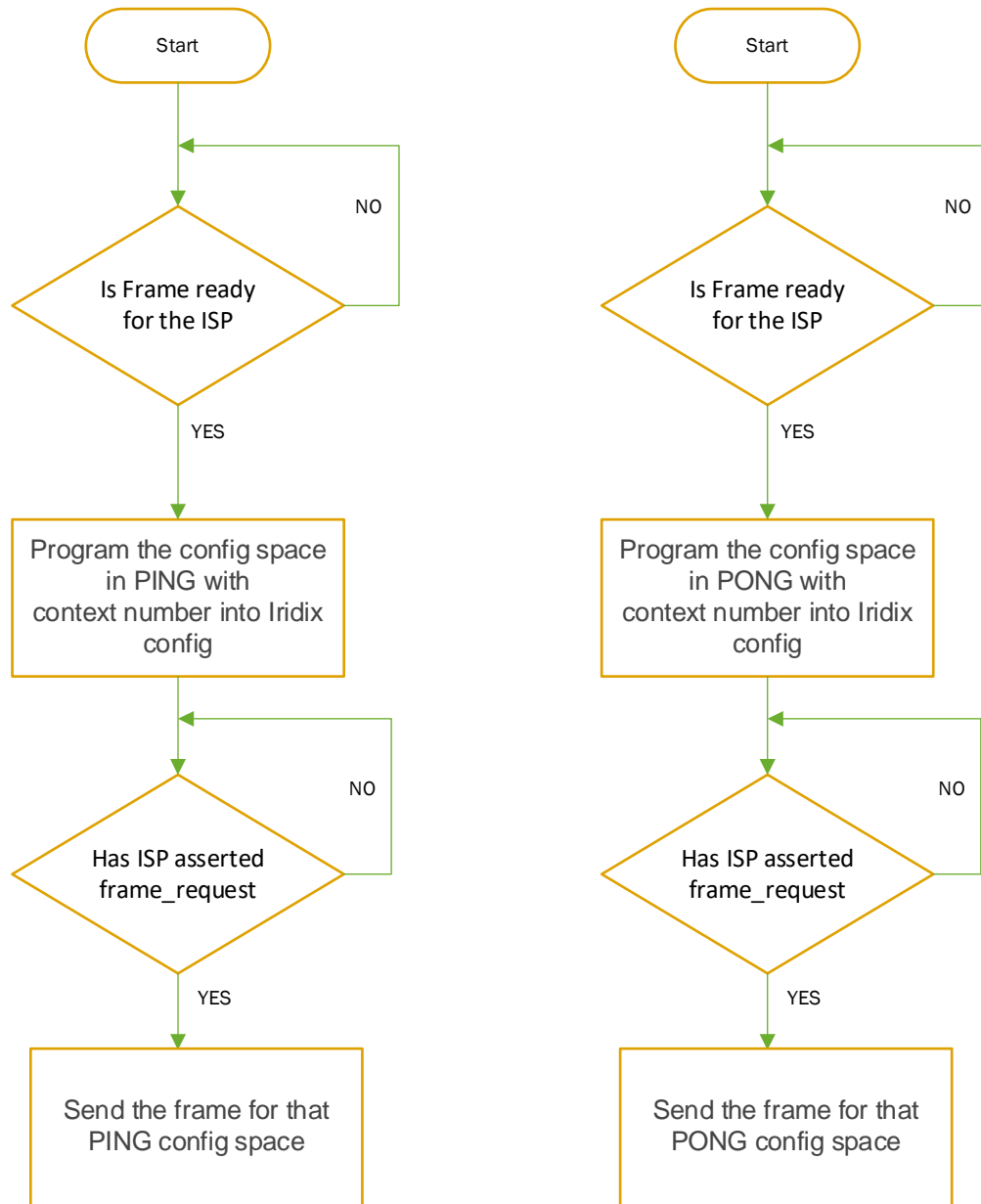


Figure 9. PING/PONG config programming in multi-context SW-HW mode

Referring to Figure 9 you can see that the flow is same for the PING and PONG address spaces.

Note: *The sequence must always be PING-PONG alternate.
The frame must be sent when ISP sends the frame request.
The ISP locks a particular config space when the corresponding frame is active within ISP.*

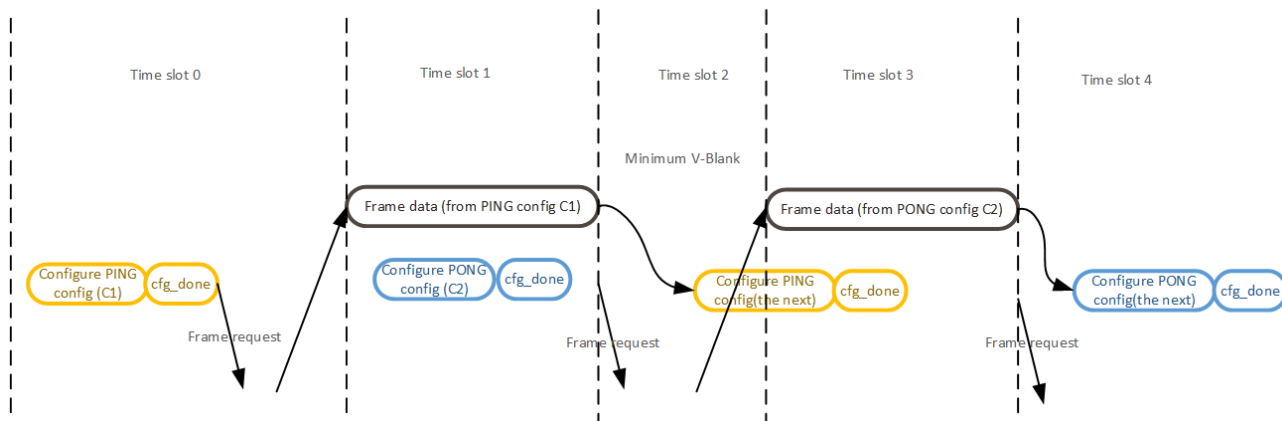


Figure 10. Multi-context SW-HW Mode

As is shown in Figure 10, the multi-context SW-HW mode is similar to the single context mode, but adds one more acknowledge mechanism to the external buffer management unit. After the hardware reset or `global_fsm_reset`, the ping pong selection always starts from PING, and keeps alternating sequentially. The related configuration for the upcoming frame must be completed before sending the data.

For example, in the time slot 0, configure PING (C1) ends ahead of the related PING frame which starts in time slot 1; while the configuration (C2) of the PONG frame which is supposed to come in the time slot 3, can start from the time slot 0 to 2 as long as it ends ahead the start of the related PONG frame in time slot 3. Similar to the single context mode, here too the ISP locks the address space which is being used currently and the other one is free to be updated.

Note: *In the multi-context SW-HW mode, if the PING/PONG config space for an upcoming frame is not configured before the first pixel arrives, then the multi-context configuration error interrupt is asserted.*

2.5 Functional assumptions

This section provides information about the requirements that must be met/followed for the correct functioning of the ISP.

- The input video data must maintain a minimum horizontal blanking of 32-pixels between two active lines. The VCKE signals must be 1 for these 32 pixels. For example, if the absolute blanking is 32 pixels, but the VCKE is 0 for 10 cycles, the effective horizontal blanking is just 22 pixels. The effective HB must be at least 32 pixels.
- Input video and the output video should not overlap. There must be minimum vertical blanking between two frames.
The minimum vblank for a:

- Full ISP configuration is 40 lines.
- ISP lite configuration is 24 lines.
- The temper module needs some data-prefetch. There should be a system DDR latency delay between the time when the DMA initiates the data prefetch and when Temper starts getting the video input data. The Sinter delay-line latency is used to ensure this delay. Therefore, the sinter CANNOT be bypassed if the temper is enabled in the multi-context mode. If the sinter functionality is not needed, but still temper is expected to be enabled, please disable sinter through ISP_CONFIG_PING:Sinter:Enable and ISP_CONFIG_PONG:Sinter:Enable signal. In the single context mode, the Sinter can also be bypassed if the ISP_COMMON:isp global chicken bit:rd start sel signal is set to 1.
- The ODD frame width is not verified. Also, the ODD frame height is NOT supported in the YUV420 mode. The last line in the frame will have incorrect data if ODD frame height is used in YUV420 mode.

2.6 Setting up the ISP

The ISP can be set up as shown in [Figure 11](#).

Note: ***By default the ISP input port is in safe_stop mode. In this mode it blocks any input frame.***

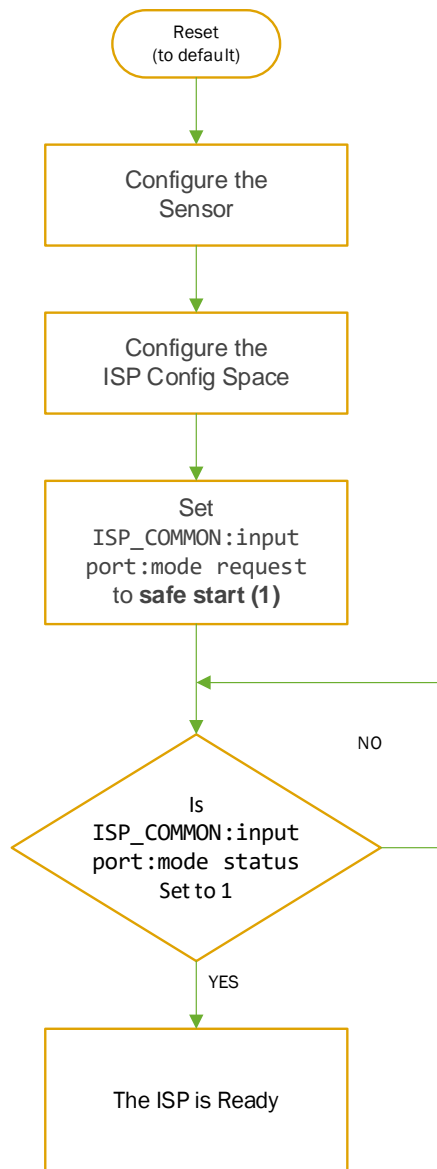


Figure 11. Setting up the ISP

2.7 Module overview

The various modules depicted in [Figure 4](#) are described in this section.

2.7.1 Input port

At the input port, preformatting is carried out on the sensor input signals to match the internal timing requirements. The preset modes described in the following section allow for

quick configuration of the input for various sensor-synchronization output formats. If a nonstandard input port configuration is required, the relevant configuration registers must be set by the user according to the needs of the sensor and type of output required.

In the following sections, the sensor horizontal sync output is referred to as the horizontal valid, or HV signal, and the vertical sync output is referred to as the frame valid or FV signal.

NOTE: *The ISP assumes that the video input has a minimum of 32 pixel clock cycles of horizontal blanking and 40/24 lines of vertical blanking. Please contact Arm support if you require smaller blanking periods.*

For more information, refer to the **INPUT PORT** section of the Mali-IV009 Register Map.

2.7.1.1 Preset input port modes

Several input port preset modes are available to synchronize with various sensors. The preset modes are as follows:

Input Port: Preset	Description
0, 1, 3, 4, 5, 7-15	Reserved.
2	FV and HV HIGH during active video.
6	FV, HV HIGH during active video with windowing capability.

2.7.1.1.1 Preset mode 2

‘Preset mode 2’ supports typical sensor sync outputs with both FV and HV signals encapsulating the active video.

The input port expects:

- Constant horizontal and vertical blanking periods
- The active video encapsulated by the FV and HV pulse.

The following figure shows these timings.

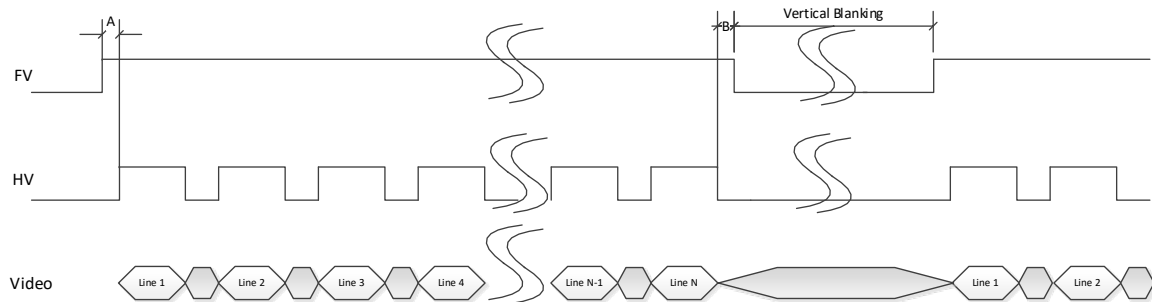


Figure 12. Input port Mode 2 timing diagram

Symbol	Definition	Min	Max	Units
A	FV rising edge to HV rising edge of first line.	0	Vertical blanking – 1.	Pixel CLK cycles.
B	HV falling edge of last line to FV falling edge.	0	100	Pixel CLK cycles.

2.7.1.1.2 Preset Mode 6

This mode supports sensors with standard FV and HV signals as in preset mode 2 and provides additional support for windowing. In this mode, the user can adjust active window registers to select a smaller window within the full active window. This may be necessary when the active video includes blanking or optical black pixels. This mode can also be used to crop the sensor active window in cases where the edge columns and rows should be excluded.

For more information, refer to the **INPUT PORT** section of the Mali-IV009 Register Map.

Figure 13 shows the use of the following settings:

Registers	Function
hc start0, hc start 1	Specifies the horizontal offset (in pixels) from the rising edge of the horizontal valid signal.
vc start	Specifies the vertical offset (in lines) from the rising edge of the frame valid signal.
hc size0, hc size1, vc size	Specifies the horizontal and vertical window sizes.

The following figure shows the dimensions that are controlled by the registers described in the previous table.

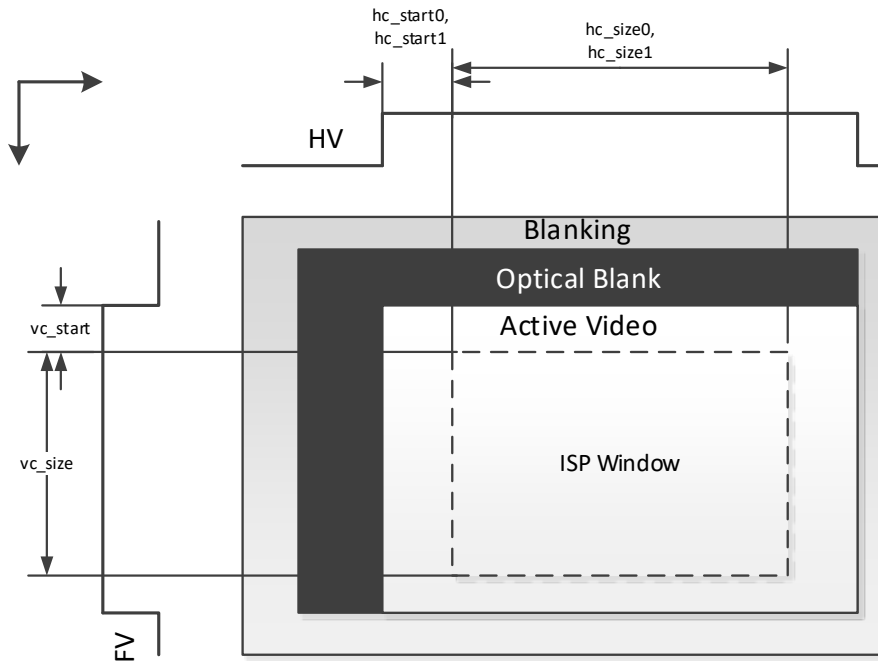


Figure 13. Input port mode 6 windowing

The following registers are used to configure this mode.

Register	Description
preset	Selector for input port preset modes.
hc start0 , hc start1	Specifies the horizontal offset in pixels in preset mode 6. These two registers should be set to the same value.
hc size0 , hc size1	Specifies the size of the horizontal window in preset mode 6. These two registers should be set to the same value.
vc start	Specifies the vertical offset in lines in preset mode 6.
vc size	Specifies the size of the vertical window in preset mode 6.
frame width	Read only value that can be used for debug. Returns the detected frame width.
frame height	Read only value that can be used for debug. Returns the detected frame height.

2.7.1.1.3 Input port start/stop

When the input timing from a sensor must be changed, either during a camera configuration change or window change, the input port can be safely stopped and started on frame boundaries using the following procedure:

1. Send a safe stop to the ISP by writing a 0 to **Input port: mode request**.
2. Read back **Input port: mode status** until the value read back is 0, indicating the ISP has stopped. Otherwise wait for the “input port safely stopped” interrupt.
3. Configure sensor registers (if required).
4. Configure ISP registers (if required).
5. Send a safe start to the ISP by writing a 1 to **Input port: mode request**.
6. Read back **Input port: mode status** until the value read back is 1, indicating the ISP has started.
7. Continue with the camera operations.

Note: *During ISP configuration changes the ISP pixel clock must be running.*

The **Input port: freeze config** bit can be set to 1 to freeze the input port configuration. This is useful when multiple register changes to the input port configuration are required. When this bit is enabled, the input port registers are shadowed and updated when the **Input port: freeze config** bit is set to 0.

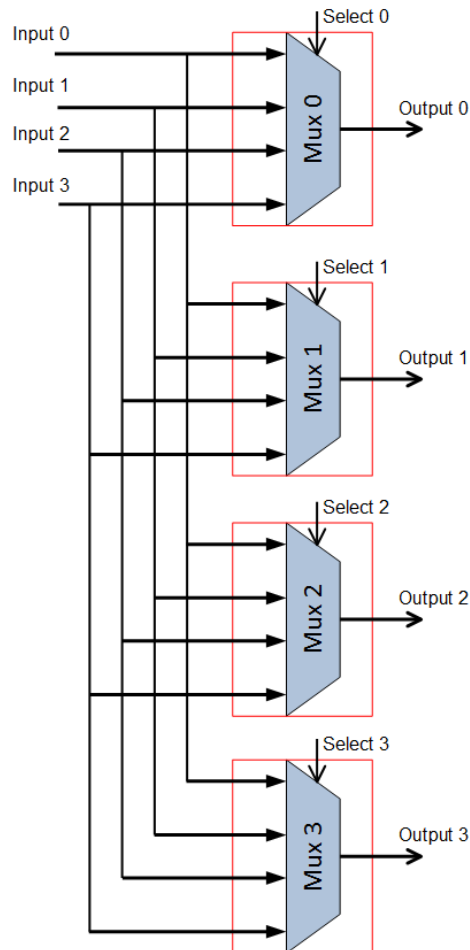
The following registers are used to configure this mode.

Register	Description
mode request	Selector for input port preset modes.
mode status	Used to monitor input port status. <ul style="list-style-type: none"> • bit 0: 1 = running, 0 = stopped • bits 1,2 = reserved
hc size0 hc size1	Specifies the size of the horizontal window in preset mode 6. These two registers should be set to the same value.
vc start	Specifies the vertical offset in lines in preset mode 6.
vc size	Specifies the size of the vertical window in preset mode 6.
freeze config	Freezes input port configuration when set to 1. Register changes on the input port are shadowed and do not take effect until this bit is set to 0.

For more information, refer to the **INPUT PORT** section of the Mali-IV009 Register Map.

2.7.2 Crossbar

The crossbar allows any-to-any mapping of the 4 video inputs to the outputs as shown in the following figure.

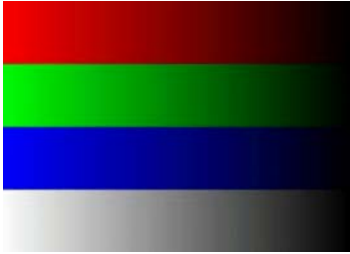

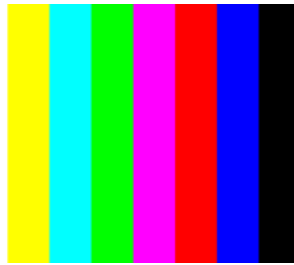



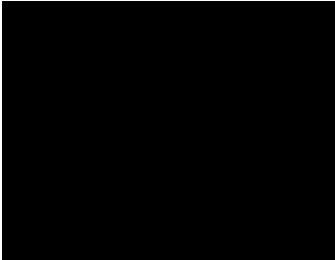
Note: Please refer to the *ISP_CONFIG_PING: crossbar* and *ISP_CONFIG_PONG: crossbar* sections in the Mali-IV009 register map for the detailed register descriptions.

2.7.3 Video test pattern generator

This block provides four instances of a flexible test pattern generator. Each of the four input channels may have a different video test pattern. This module is especially useful at the start of a project, because it enables the ISP signals to be tested and may be useful in testing for connectivity during production testing. Valid clock and frame geometry must be provided by the input module.

The test pattern generator can replace the video data with a test pattern selected from the following:

Pattern	Values																																				
Flat field	All pixels are set to video test gen: r backgnd , video test gen: g backgnd , video test gen: b backgnd values.																																				
Horizontal gradient 	<p>Lines are colored as follows:</p> <ul style="list-style-type: none">• Top 25% of lines are red.• Next 25% of lines are green.• Next 25% of lines are blue.• Bottom 25% of lines are grey. <p>Left is light, with initial value set using rgb gradient start <i>n</i>, where <i>n</i> is the number of the channel.</p> <p>Right is dark. Color increment per pixel set using rgb gradient <i>n</i>, where <i>n</i> is the number of the channel.</p>																																				
Vertical gradient 	<p>Pixels are colored as follows:</p> <ul style="list-style-type: none">• Left 25% of pixels are red.• Next 25% of pixels are green.• Next 25% of pixels are blue.• Right 25% of pixels are grey. <p>Top is dark, with initial value set using video test gen: rgb gradient start <i>n</i>, where <i>n</i> is the number of the channel.</p> <p>Bottom is light. Color increment per pixel set using video test gen: rgb gradient <i>n</i>, where <i>n</i> is the number of the channel.</p>																																				
Vertical bars 	<p>Eight vertical bars of color, organized from left to right as follows (all registers from the group video test gen):</p> <table><tr><th>Color</th><th>Red</th><th>Green</th><th>Blue</th></tr><tr><td>White (Grey)</td><td>r backgnd</td><td>g backgnd</td><td>b backgnd</td></tr><tr><td>Yellow</td><td>r backgnd</td><td>g backgnd</td><td>0</td></tr><tr><td>Cyan</td><td>0</td><td>g backgnd</td><td>b backgnd</td></tr><tr><td>Green</td><td>0</td><td>g backgnd</td><td>0</td></tr><tr><td>Magenta</td><td>r backgnd</td><td>0</td><td>b backgnd</td></tr><tr><td>Red</td><td>r backgnd</td><td>0</td><td>0</td></tr><tr><td>Blue</td><td>0</td><td>0</td><td>b backgnd</td></tr><tr><td>Black</td><td>0</td><td>0</td><td>0</td></tr></table>	Color	Red	Green	Blue	White (Grey)	r backgnd	g backgnd	b backgnd	Yellow	r backgnd	g backgnd	0	Cyan	0	g backgnd	b backgnd	Green	0	g backgnd	0	Magenta	r backgnd	0	b backgnd	Red	r backgnd	0	0	Blue	0	0	b backgnd	Black	0	0	0
Color	Red	Green	Blue																																		
White (Grey)	r backgnd	g backgnd	b backgnd																																		
Yellow	r backgnd	g backgnd	0																																		
Cyan	0	g backgnd	b backgnd																																		
Green	0	g backgnd	0																																		
Magenta	r backgnd	0	b backgnd																																		
Red	r backgnd	0	0																																		
Blue	0	0	b backgnd																																		
Black	0	0	0																																		

Pattern	Values
Arbitrary rectangle 	Flat field background color = (video test gen: r backgnd, video test gen: g backgnd, video test gen: b backgnd). Rectangle foreground color = (video test gen: r foregnd, video test gen: g foregnd, video test gen: b foregnd). Rectangle pixel geometry = (rect top, rect bot, rect left, rect right) (first line = 1, first pixel = 1).
Default pattern 	2 pixel white (100% RGB) frame on black background (0% RGB).

Note: *Please refer to the **ISP_CONFIG_PING: video test gen ch0 to ch3** and **ISP_CONFIG_PONG: video test gen ch0 to ch3** sections in the Mali-IV009 register map for the detailed register descriptions.*

The following registers are used to configure this block:

Register	Description
test_pattern_off_on	Test pattern enable. 0 = off, 1 = on.
pattern type	8-bit selector for test pattern: 0 = Flat field. 1 = Horizontal gradient. 2 = Vertical Gradient. 3 = Vertical Bars. 4 = Arbitrary Rectangle. 5 = Random. 6 – 255 = Default white frame on black.
r backgnd g backgnd b backgnd	R, G, B and IR values for background in Flat field, Vertical bars and Arbitrary rectangle test patterns.
r foregnd , g foregnd b foregnd	R, G, B and IR values for windowed rectangle in Arbitrary rectangle test pattern.
rgb gradient start <n>	Specifies the starting brightness value of the gradient in Vertical and Horizontal gradient test patterns for channel n .
rgb gradient <n>	16-bit value specifying the per pixel increment in the gradient for channel n in 4.12 fixed point format.
rect top rect bot rect left rect right	Specifies the coordinates of the windowed rectangle in Arbitrary rectangle test pattern. Note: (1, 1) corresponds to the first pixel.

2.7.4 Input formatter

When companded data is received from the sensor(s), this module is responsible for decompanding the raw video, if required. The decompanding modes include:

- Knee-point (PWL) decompanding.
- Log⁻¹ decompanding.
 - Approximating the function $y = \text{round}(\text{pow}(2.0, x/4096.0) * 16.0)$ where x is a 16-bit number and y is a 20-bit number. This function is used by OmniVision's sensors.
- Reconstruction of exposures where sensors transmit only two of three possible HDR exposures.

These modes may be used only singly but can be selected individually for each frame.

When the knee point and slope values have been set, the ISP selects the appropriate slope as follows:

- The value of **knee_point0** must be less than or equal to **knee_point1** *AND* The value of **knee_point1** must be less than or equal to **knee_point2**.
- If the input data value is less than or equal to **knee_point0**:
slope0 is used.
- If the value of **knee_point0** is less than the input data value *AND* If the value of the input data value is less than or equal to **knee_point1**:
slope1 is used.
- If the value of **knee_point1** is less than the input data value *AND* If the value of the input data value is less than or equal to **knee_point2**:
slope2 is used.
- If the value of the input data is greater than **knee_point2**:
slope3 is used.

The following registers are used to configure this part of the pipeline. These registers are all members of the **Input formatter** group.

Register	Description																
mode in	1 = 2:3 multiple exposure multiplexing 2 = Logarithmic encoding 3 = Companding curve with knee points 4 = 16bit linear+ 12bit VS 5 = 12bit companded + 12bit VS 6 = reserved 7 = pass through mode																
input bitwidth select	0 = 8-bits. 1 = 10-bits. 2 = 12-bits. 3 = 14-bits. 4 = 16-bits. 5 = 20-bits (no 18-bits setting).																
knee point<n> (0 to 2)	Knee point values are provided by the sensor manufacturer.																
Slope<n> select (0 to 3)	<table><tr><td>0 = 1x</td><td>1 = 2x</td><td>2 = 4x</td><td>3 = 8x</td></tr><tr><td>4 = 16x</td><td>5 = 32x</td><td>6 = 64x</td><td>7 = 128x</td></tr><tr><td>8 = 256x</td><td>9 = 512x</td><td>10 = 1024x</td><td>11 = 2048x</td></tr><tr><td>12 = 4096x</td><td>13 = 8192x</td><td>14 = 16384x</td><td>15 = 32768x</td></tr></table>	0 = 1x	1 = 2x	2 = 4x	3 = 8x	4 = 16x	5 = 32x	6 = 64x	7 = 128x	8 = 256x	9 = 512x	10 = 1024x	11 = 2048x	12 = 4096x	13 = 8192x	14 = 16384x	15 = 32768x
0 = 1x	1 = 2x	2 = 4x	3 = 8x														
4 = 16x	5 = 32x	6 = 64x	7 = 128x														
8 = 256x	9 = 512x	10 = 1024x	11 = 2048x														
12 = 4096x	13 = 8192x	14 = 16384x	15 = 32768x														

Note: *For more information about the registers refer to the **ISP_CONFIG_PING:input formatter and ISP_CONFIG_PONG:input formatter in the Mali-IV009 register map.***

2.7.5 Sensor offset WDR and gain WDR

In some circumstances an offset, gain or both may need to be applied to the data before frame switching and frame stitching. Different gain and offset values may be applied to each of the *Wide Dynamic Range* (WDR) channels.

The offset values are provided for each colour plan and each channel. For a given exposure, the black levels offsets for a plane is the delta from the smallest black level of that exposure. At the end of this operation, all the colour planes for a given exposure will have same black level

Note: *This operation does not remove the black level. It just balances the black level difference between the colour planes*

2.7.6 Frame stitch

This module allows up to four exposures to be combined into a single WDR image in 20-bit linear space. The algorithm allows alpha blending at exposure transition points and is also motion adaptive, which greatly reduces the artefacts associated with line overlapped multi-exposure WDR/HDR.

When motion is detected, according to a configurable threshold, bits can be set in the pixel consistency plane. This is an additional image plane, output through DMA, containing flags indicating the reliability of pixels. This is intended as an informative layer for computer-vision algorithms.

The purpose of WDR exposure stitching is to combine two or more temporally-proximal frames of different exposure length into a single frame of greater precision than the original frames.

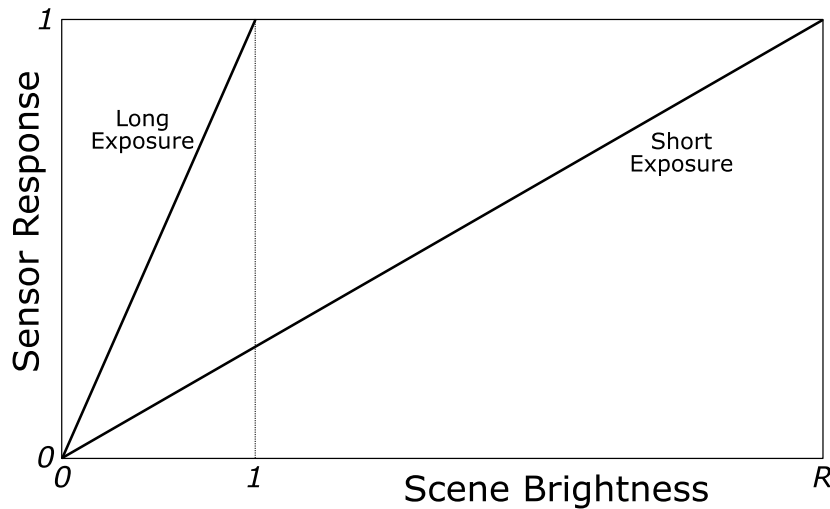


Figure 14. Intensity response for two exposures

The graph above shows the intensity response for two exposures, one long and one short. It is assumed that the sensor has a linear response, and thus the ratio R , between integration times is the same as the ratio between the gradients of the responses.

To create a simple stitch of these two exposures, we first divide the long exposure by the exposure ratio, R , to align the two exposures. Then for each pixel we select between the two exposures by comparing the value of the longer exposure against a threshold t :

$$t < \frac{1}{R}.$$

For values below the threshold we select the (scaled) long exposure, and for values above we select the short exposure. Thus, we now have an image in which all of the darker regions are from the longer exposure (and so are less noisy), and all of the brighter regions are from the shorter exposure (and so less likely to be clipped). In practice, to avoid artefacts caused by transitioning suddenly between pixels from different exposures, we select two thresholds, t_1 and t_2 , and alpha-blend between the two exposures as follows.

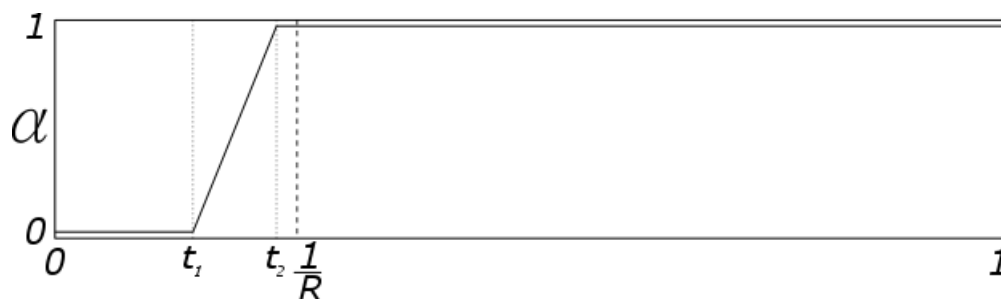


Figure 15. Alpha blending two exposures

Two major issues remain. Firstly, the raw data is Bayer data, so the intensity of a pixel intensity depends on local color. The higher the saturation of a region, the greater the disparity between pixel values from different color channels. Selecting pixels based purely on their values will result in pixels in saturated regions being selected from different exposures depending on their channel. For example, green from the short exposure and red and blue from the long exposure. This will lead to color artefacts in the image after the demosaic process. To avoid this, we must ensure that pixel selection is based on intensity within a small region around the pixel. One way to achieve this is to dilate the raw data with a 3 x 3 max filter before generating the alpha map. Another is to generate the alpha map from the data, then dilate the alpha map with a 3 x 3 filter.

The second major issue is motion. When there is motion in the scene, or the camera is moving, there will be disparity between consecutive exposures. In this case pixels at the same location in different exposures may have very different values after scaling has been applied. Applying the simple stitching method described above will result in image artefacts such as discontinuities and objects being repeated.

2.7.6.1 Detecting motion

Motion can be detected by examining the differences between pixel values once exposures have been scaled according to the exposure ratio. However, several factors must be taken into account. Firstly, both exposures will have some degree of noise. Once the exposures have been scaled, the noise in the short exposure will be greater than that of the long exposure. Because of this noise, corresponding pixels from the two scaled exposures will have different values, even in the absence of motion. We estimate this difference using the noise profile.

The registers used to provide the optimum response to motion are as follows:

Register	Exposure pair	Typical value
frame stitch: LM NP mult	Long / Medium exposures.	384
frame stitch: MS NP mult	Medium / Short exposures.	1 536
frame stitch: SVS NP mult	Short / Very Short exposures.	1 536

Table 3. WRD stitching

Note: *Please refer to the **ISP_CONFIG_PING: frame stitch** and **ISP_CONFIG_PONG: frame stitch** sections in the **Mali-IV009 register map** for the detailed register descriptions.*

Higher values in these registers make the ISP less sensitive to both noise and motion.

Secondly, in areas of motion, the long exposure may exhibit motion blur. This may cause the difference between the exposures to be less than it would otherwise be. If the difference is lower than the expected noise level, some or all of the motion will not be detected.

Finally, differences between values of corresponding pixels will be affected by clipping. Specifically, in the absence of motion, regions which are so bright that the long exposure is clipped, may have short exposure values anywhere above the clipping point of the long exposure. In regions where the long exposure is clipped, it is only possible to detect motion when the short exposure has a value lower than that of the long exposure. In such cases, we can modify the motion ramp equation as follows:

$$\alpha_{mov} = \min \left(1, \max \left(0, \left((x_l - x_s) - n(x_l - x_s) \right) P_M \right) \right)$$

2.7.6.2 Dealing with motion

If motion is detected, one of two methods may be used to modify the alpha map used to blend the two exposures:

1. Always use the short exposure in regions of motion.
2. Always use the long exposure in regions of motion.


The advantage of using the short exposure in regions of motion is that it is less likely to be clipped. It is also not necessary to detect motion any differently in regions where the long exposure is clipped, as the short exposure will be used here in either case. The disadvantage is that the short exposure will have much more noise. In darker areas, the short exposure may be so noisy that there is no real detail present so this approach will lead to noisy, black regions.

Conversely, the advantage of using the long exposure is that it is much less noisy. However, clipped pixels must be dealt with in two ways. Firstly, when detecting motion, we must take into account the fact that they are clipped by modifying the motion ramp equation as described above. Failure to do so will result in sufficiently bright regions always being confused for motion. Secondly, if these clipped pixels are used directly, they will result in the final image having erroneous colors - usually a shift towards magenta.

To understand why this is, imagine a very bright region in which all three color channels are clipped in the long exposure. Later in the pipeline, white balance correction will be applied, by multiplying the red and blue channels by some coefficients (greater than one). The result will be a magenta color where we expect to see white or light grey. We may account for this by pre-empting the white balance correction, and increasing the green channel accordingly. Though we can correct for the shift towards magenta, it is not always possible to recover the true color of the region - the information lost due to clipping may not be recovered.

Each of these two strategies has a penalty for false positives in motion detection. The first will introduce extra noise by relying on the short exposure. The second will remove detail and potentially introduce false colors by using clipped pixels from the long exposure.

The following registers are used to configure this feature:

Register	Description
Cfa pattern (Top)	The pixel arrangement of the CFA array on the sensor. Set in the Top register group and used by several blocks in the pipeline: 0 = RGGB. 1 = RCCC. 2 = RlRGB. 3 = RGlRB.
RGGB start pre mirror (Top)	 Position of the R cell in the four pixel group.
frame stitch: mode in	Specifies the number of exposures input: 0 = 4 exposures. 1 = 2 exposures. 2 = 3 exposures. 3 = 4 exposures.
frame stitch: * Thresh high	Data above this threshold will be taken from shorter of two exposures. Must be greater than corresponding frame stitch: * Thresh low .
frame stitch: * Thresh low	Data below this threshold will be taken from longer of two exposures. Must be lower than corresponding frame stitch: * Thresh high .
frame stitch: * Exposure Ratio	Sets the ratio between a pair of exposures. This must match the actual exposure ratio on the sensor. In practice this will never be higher than 1024.
frame stitch: Black level Long	Black level for long exposure input. The value will be around 3 840 for 16-bit input.
frame stitch: Black level Medium	Black level for the medium exposure input. The value will be around 3840 for 16-bit input.
frame stitch: Black level Short	Black level for the short exposure input. The value will be around 3 840 for 16-bit input.
frame stitch: Black level very Short	Black level for the very short exposure input. The value will be around 3 840 for 16-bit input.
frame stitch: Black level Out	Black level for stitched output. The value will be around 983 040 for 24-bit output.
frame stitch: * Alpha MOV slope	This defines the gradient of the motion alpha ramp. Higher values indicate a steeper ramp and so a more rapid transition between non-motion-corrected and motion-corrected regions.

Note: Please refer to the *ISP_CONFIG_PING: top*, *ISP_CONFIG_PONG: top*, *ISP_CONFIG_PING: frame stitch*, and *ISP_CONFIG_PONG: stitch* sections in the Mali-IV009 register map for the detailed register descriptions.

2.7.7 Decompander0 and Decompander1

For more information about the registers refer the Mali-IV009 register map.

This module provides linear expansion followed by dynamic range compression used for high dynamic range image sensors with a compressed output. The lower bit-width sensor output is typically a piecewise linear compressed representation of the linear video covering up to a 20-bit number range. This Pre-compander module functionally converts the sensor values back into a linear range and applies dynamic range compression (DRC) to reduce the values back to 12 bits for processing in the ISP. These two operations are implemented using two LUTs.

The Pre-companding LUTs, LUT0 (decompander0) and LUT1 (decompander1), should be programmed to match the knee points in the sensor and the target output profile that will be compatible with the AWB algorithm in order to maintain color accuracy. Please contact Arm support for reference curves, LUT generation code, and AWB reference code.

In typical configurations, LUT0 has 33 evenly spaced nodes labeled 0...32 and LUT1 has 257 evenly spaced nodes labeled 0...256 with linear interpolation applied between these nodes. Each data value is a 16 bit unsigned number. Each LUT is implemented as an array of 32 bit registers where the 12 MSbits are set to 0.

Note: *Modification of the LUT during active video causes corrupted video. To avoid this, the writing to LUT must be done during the vertical blanking period.*

2.7.7.1 Key parameters

Parameter	Description
Enable	Set to 1 enable WDR Pre-companding LUT.
LUT0_Data(0)...Data(32)	33, 20 bit LUT values
LUT1_Data(0)...Data(256)	257, 20 bit LUT values

2.7.8 Digital gain

This module applies a gain in 5.8 format to all the bayer colour planes (R,Gr,Gb,B). The gain is applied in the linearized data before compressing.

The offset is subtracted before the gain is applied and then the offset is added back to the data.

Register	Description
Gain	Gain applied to data. 5.8 format.
Offset	Data black level. 20-bit value.

Note: *Please refer to the ISP_CONFIG_PING: digital and ISP_CONFIG_PONG: digital gain sections in the Mali-IV009 register map for the detailed register descriptions.*

2.7.9 Front end sensor offset

This module provides correction for the black level from the sensor data. Separate offsets are provided for each of the 4 Bayer component color channels (R, Gr, Gb, B).

This module must operate only in the non-HDR raw data format. The offset values are provided for each colour plan. The black levels offsets for a plane is the delta from the smallest black level of all the colour planes. At the end of this operation, all the colour planes will have the same black level.

Note: *This operation does not remove the black level. It just balances the black level difference between the color planes.*

The following registers are used to configure this block.

Register		Description
Digital Gain	Gain	Gain applied to data. 4.8 format.
	Offset	Data black level. 20-bit value.
Sensor Offset fe	offset 00	Black offset subtraction for each channel in linear domain: Channel 00 (R).
	offset 01	Black offset subtraction for each channel in linear domain: Channel 01 (Gr).
	offset 10	Black offset subtraction for each channel in linear domain: Channel 10 (Gb).
	offset 11	Black offset subtraction for each channel in linear domain: Channel 11 (B).

Note: *Please refer to the ISP_CONFIG_PING: digital gain, ISP_CONFIG_PONG: digital gain, ISP_CONFIG_PING: sensor offset fe, and ISP_CONFIG_PONG: sensor offset fe sections in the Mali-IV009 register map for the detailed register descriptions.*

2.7.10 Square Root

The ISP supports a maximum linear data width of 20bits. But the data is compressed to a maximum of 16bits before any noise reduction happens. A square root function is used to compress the data.

Before the square root function, the black level is subtracted in the linear domain and then the black level is added back in the square root domain.

Note: *Please refer to the **ISP_CONFIG_PING: sqrt** and **ISP_CONFIG_PONG: sqrt** sections in the Mali-IV009 register map for the detailed register descriptions.*

2.7.11 Raw frontend

The Raw frontend (RFE) module is responsible for green equalization and dynamic defect pixel correction.

For more information about the related registers, refer to the Mali-IV009 Register Map.

2.7.11.1 Green equalization

Green equalization corrects imbalances between pixels in the Gr and Gb color planes, and adjusts the planes to have equal sensitivity. This correction reduces the appearance of checker patterns after the demosaic process and can also improve false color correction in the demosaic algorithm.

Green pixels are compared with the average of the neighbouring Gr and Gb values. If the differences are smaller than the value of **RAW Frontend: ge threshold** the pixels are replaced with a blended value based on the raw pixel value and the neighbouring differences between the two green channels. The **RAW Frontend: ge slope** register is used to define the amount of blending for differences above **RAW Frontend: ge threshold**. Increasing **RAW Frontend: ge threshold** or **RAW Frontend: ge slope** will result in more aggressive correction. The diagram below shows a curve representing the amount of blending as a function of the difference between the two green channels.

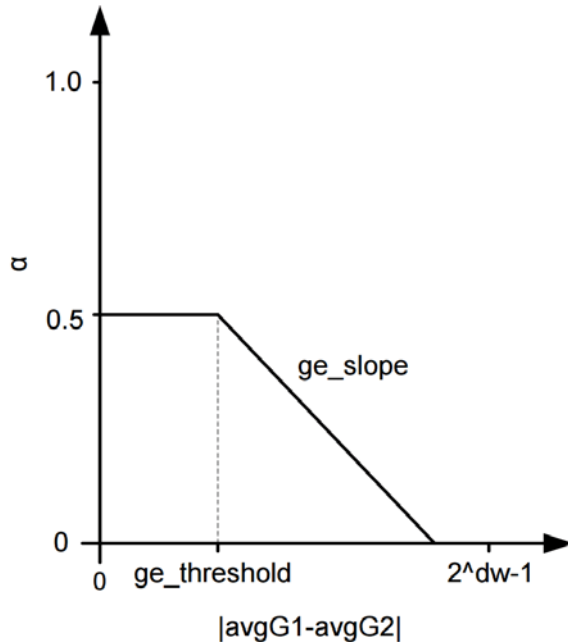


Figure 16. Gamma_GE blending threshold

As correction of the greens should not take place on edges, an additional register, **RAW Frontend: ge sens**, is used to adjust sensitivity of the green equalization to detected edges. A higher **RAW Frontend: ge sens** value will result in more aggressive green equalization on edges.

The following registers are used to configure this mode. These registers are all members of the **RAW Frontend** group:

Register	Description
ge enable	Green equalization enable. 0 = off, 1 = on.
ge strength	A register set during calibration.
ge threshold	Controls the strength of green equalization. Higher values result in stronger green equalization.
ge slope	Controls the strength of blending for green imbalances larger than RAW Frontend: ge threshold . Higher values result in stronger green equalization.
ge sens	Controls the sensitivity of green equalization to edges. Higher values result in stronger green equalization on detected edges.

Note: *Please refer to the **ISP_CONFIG_PING: raw frontend** and **ISP_CONFIG_PONG: raw frontend** sections in the Mali-IV009 register map for the detailed register descriptions.*

2.7.11.2 Dynamic defect pixel correction

Dynamic Defect Pixel (DDP) performs on-the-fly detection and correction of both bright and dark defective pixels in the raw Bayer pattern domain. There is no limitation on the number of single pixel or small cluster defects which can be corrected. If the pixel input is detected as a defect pixel, this block replaces the raw pixel input value with a corrected pixel value. This method is generally less reliable than the static method in scenes where details approach the Nyquist frequency.

DDP employs a 5 x 5 kernel function to both detect defect pixels and apply correction. DDP is mostly applicable to display applications to improve the appearance of images, removing single and small pixel clusters of defects. The correction logic, which includes false color suppression, may be disabled independently of the detection logic. In this case, the results of the detection logic continue to be available as a defect pixel count and as a map in the pixel consistency plane.

2.7.11.2.1 Pixel replacement values

Both directional and non-directional replacement blocks are used to calculate intermediate directional and non-directional replacement values.

The non-directional logic block provides a non-directional replacement value based on the median (green pixels) or neighbourhood average (red and blue pixels).

The directional replacement logic block provides a directional replacement value which is biased to preserve detected edges. The **RAW Frontend: line thresh** register can be used to adjust the directional bias of the directional replacement value. Smaller values for the **RAW Frontend: line thresh** will make the directional replacement values more sensitive to edges and may result in improved resolution in high spatial-frequency regions.

The final replacement value is a blended result of the directional and non-directional replacement values where blending is based on the **RAW Frontend: dp blend** register. Higher **RAW Frontend: dp blend** values will bias the blending towards the directional replacement value. The blending is linear where a value of 0 results in the final replacement value being the non-directional replacement value. A value of 255 results in the final replacement being the directional replacement value.

2.7.11.2.2 Dynamic defect pixel detection and replacement

The determination of whether a pixel is defective is based on the standard deviation of the neighboring pixels and the deviation of the raw pixel input from the median of its neighboring pixels of the same color. The Detection Logic block generates a mask or alpha blending parameter, α_{out} , based on the calculated deviation and the **RAW Frontend: dp threshold** and **RAW Frontend: dp slope** registers as shown in the following figure.

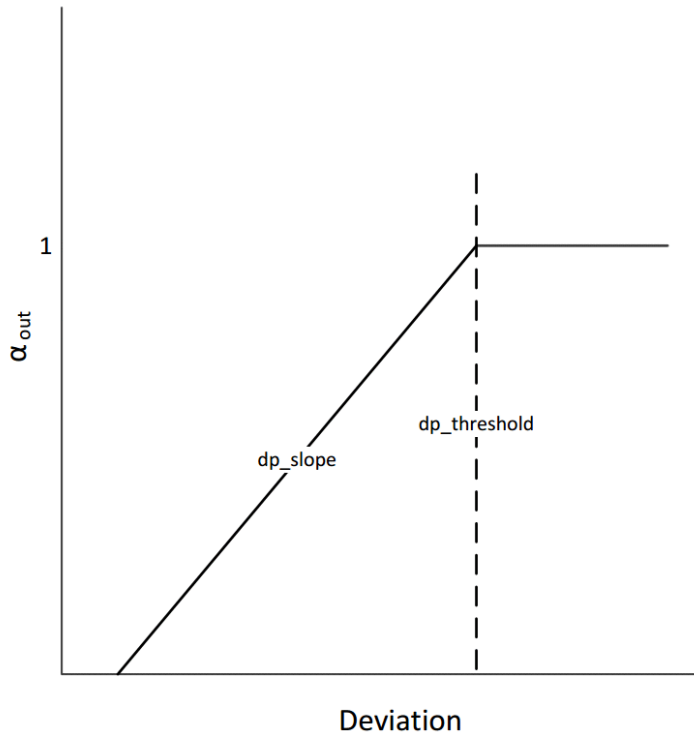


Figure 17. Blending logic curve

Pixels with deviation above the **RAW Frontend: dp threshold** are replaced with the final replacement values. Pixels with deviation below the **RAW Frontend: dp threshold** are alpha blended with the original pixel input value and the final replacement value. The degree of blending between the original pixel value and the final replacement value depends on the **RAW Frontend: dp slope** register. Higher values of **dp slope** will increase the amount of blending between the original pixel value and the final replacement value. A value of zero for **RAW Frontend: dp slope** would cause all pixels with deviation lower than **RAW Frontend: dp threshold** to be uncorrected.

Note: *In the presence of a defective pixel, green equalization is disabled.*

The following registers are used to configure this mode. These registers are all members of the **RAW Frontend** group:

Register	Description
ge enable	Green equalization enable: 0 = off, 1 = on.
dp enable	Dynamic defect pixel correction enable. 0 = off, 1 = on.

Register	Description
show dynamic defect pixel	Show Defect Pixel: 0 = off, 1 = on.
dark disable	Disable detection of dark pixels.
bright disable	Disable detection of bright pixels.
dp threshold	8-bit threshold value for dynamic pixel detection. Lower values result in more aggressive correction.
ge threshold	8-bit threshold value for green equalization.
dp slope	8-bit value which controls the aggressiveness of the alpha blending between original pixel values and the filtered pixel value for pixels below the RAW Frontend: dp threshold . Higher values result in more aggressive correction.
ge slope	Slope for the green equalization mask function.
dpdev threshold	Controls the aggressiveness of the correction to control false colors near edges. Lower values may reduce colored edges due to overcorrection, but may allow more uncorrected defect pixels.
ge sens	Controls the sensitivity of green equalization to edges.
line thresh	Controls the directional nature of the directional replacement algorithm. Higher values reduce the directional nature of the replacement value.
dp blend	Controls blending between non-directional and directional replacement values in dynamic defect pixel correction. 0x00 Replace detected defects with non-directional replacement value 0xFF Replace detected defects with directional replacement value.

Note: *Please refer to the **ISP_CONFIG_PING: raw frontend** and **ISP_CONFIG_PONG: raw frontend** sections in the Mali-IV009 register map for the detailed register descriptions.*

2.7.12 Static defect pixel correction

For more information about the registers refer to the **DEFECT PIXEL** and the **RAW FRONTEND** sections of the Mali-IV009 register map.

For static defect pixel correction, a map of defective pixels is generated using a self-calibration procedure. The values of locations specified in the defect pixel map are replaced by the median of neighbouring pixels. The total number of defect pixels that can be corrected by the static defect pixel correction is limited to the amount of memory available to store the defect pixel locations.

During the calibration procedure, the static defect pixel locations are determined using the **RAW Frontend: dp threshold** and **RAW Frontend: dp slope** registers. The values of these registers are shared with the dynamic defect pixel correction block during the static

defect pixel calibration procedure. These registers do not affect static defect pixel correction during normal operation.

Like dynamic defect pixel correction, a pixel is determined to be a defect pixel based on the standard deviation of the neighbouring pixels and the deviation of the pixel value from the median of its neighbouring pixels of the same color. The static defect pixel threshold based on **RAW Frontend: dp threshold** and **RAW Frontend: dp slope** is shown in [Figure 18](#).

During correction, all pixels with a deviation value greater than the x-intercept are considered static defect pixels and are replaced with the median of their neighbours of the same color. Unlike dynamic defect pixel correction, there is no alpha blending of original pixel values.

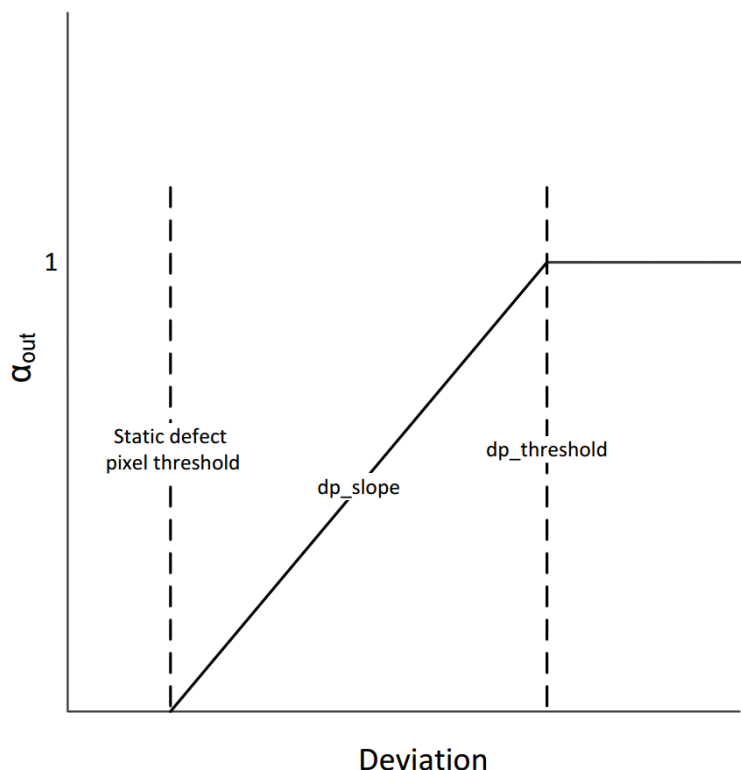


Figure 18. Static defect pixel threshold

The locations of defect pixels are stored in local memory and can be read back and stored in external non-volatile memory during calibration. When the system is powered-up the defect pixel map can be written back to the ISP memory. Reference code is available for the typical calibration process which involves adjusting the defect pixel threshold until a certain number of hot pixels are found. This corrects the worst-case defect pixels in the image with the static correction. The procedure is as follows:

1. Program the sensor for slow frame rate to increase exposure and intensity of hot pixel defects.
2. Program the analog and digital gains to their minimum values so that sensor noise is minimized.
3. Close the Iris (or place a cap over the lens).
4. The MCU takes control of the config switch through `mcu_override_config_select`. Refer to the **ISP_COMMON: isp global** section of the Mali-IV009 Register Map.
5. The MCU selects the ping config through `mcu_ping_pong_config_select`.
6. Set the **RAW Frontend: dp threshold** and **RAW Frontend: dp slope** according to the average image sensor characteristics.
7. Program the `Defect_pixel_count_in` register with the value 0. Please refer to defect pixel register description in register map.
8. Program the `Detection_enable` register with the value 1.
9. Program the `Correction_enable` register with the value 0.
10. Enable raw frontend and defect pixel.
11. Run this for a few frames.
12. Program the `Detection_enable` register with the value 0.
13. Read the `defect_pixel_count` and `table_start` register values.
14. Read the `defect_pixel_count` number of locations from the DCP ping memory starting from `table_start` value.
15. Store the content of the memory to separate locations as future tables.

In the normal operation, the correction process is automatic if the defect pixel correction register is enabled. The static defect pixel correction process is as follows:

1. Program the `Detection_enable` register with the value 0.
2. Program the `Correction_enable` register with the value 1.
3. Program `defect_pixel_count_in` with the value received in the detection stage.
4. Program the table with the content received in the detection stage.
5. Start the programming from 0th location in memory.

The following registers are used to configure this process.

Register	Description
Correction Enable	Set to 1 to enable static defect pixel correction. Contents of defect pixel LUT must be valid values.
Show Reference	Displays reference image which is compared with actual values to detect bad pixels. This should be viewed with the iris closed or with the lens capped. Note that the reference image will continue to be processed unless the subsequent ISP blocks are bypassed.
Show Static Defect Pixels	Shows detected static pixels by setting the detected pixel location to the maximum intensity. This may be seen as colored pixels if the subsequent ISP blocks are not bypassed.
Detection Enable	Setting this bit to 1 starts the hot pixel calibration procedure. This must be set back to 0 to stop the calibration.
dp threshold (RAW Frontend)	8-bit threshold value used for static defect pixel calibration. This register is shared with dynamic defect pixel correction and is used only during the static defect pixel calibration procedure.
dp slope (RAW Frontend)	8-bit value which used for static defect pixel calibration. This register is shared with dynamic defect pixel correction and is used only during the static defect pixel calibration procedure.
Defect Pixel Count	Read only value indicating the number of static hot pixels.
Defect Pixel Count In	The number of static hot pixels to be written after writing to the defect pixel LUT.
Table Start	Defect pixel LUT address used to access hot pixel locations stored in the defect pixel LUT.

Note: *Please refer to the **ISP_CONFIG_PING: raw frontend**, **ISP_CONFIG_PING: defect pixel**, **ISP_CONFIG_PONG: raw frontend**, and **ISP_CONFIG_PONG: defect pixel** sections in the **Mali-IV009 register map** for the detailed register descriptions.*

2.7.13 Sinter (special noise reduction)

Sinter® is an advanced spatial noise reduction module combining a set of algorithms to suppress sensor noise while preserving edges and image textures.

Two versions of Sinter®, version 2.5 and version 3.0, are available. A particular version can be selected by using appropriate parameter settings. (Refer to the parameter **ARM_ISP_LITE_ENABLE** in Table 1.). For more information about the Sinter® versions please contact Arm support.

This Sinter® is controlled by numerous control registers. For more information about the registers refer the Mali-IV009 register map.

The use of this module is simplified by using an externally-generated sensor Noise Profile LUT.

When this LUT is correctly programmed, the module is controllable through a reduced set of registers. In most cases only the Sinter threshold is modified to adjust noise filter strength. **Sinter: Thresh Long** and **Sinter: Thresh Short** registers corresponding to the long and short exposures respectively are used in WDR mode. When the exposure ratio is 1 or when WDR is disabled **Sinter: Thresh Long** and **Sinter: Thresh Short** should be set to the same value. The threshold values are determined during a standard calibration procedure using images captured at various ISO values and are modulated with system gain. If the image has been created using frame-switching, these values should be set accordingly based on the exposure ratio.

Note: *Please refer to the **ISP_CONFIG_PING: raw frontend**, **ISP_CONFIG_PING: sinter**, **ISP_CONFIG_PONG: raw frontend**, and **ISP_CONFIG_PONG: sinter** sections in the Mali-IV009 register map for the detailed register descriptions.*

2.7.13.1.1 Radial Sinter strength

When radial shading is applied, Sinter needs to adjust the noise correction.

Radial shading can boost the luminance of pixels affected by sensor shading and/or lens shading artefacts. However, this can exacerbate noise in these regions.

Within the Sinter core there is the option to variably increase the strength of the multi-scale filter thresholds, if the current pixel being processed is within a programmable radial distance from a programmable centre point of the image.

2.7.13.2 Lens shading

In cases where shading is used it is possible to vary Sinter thresholds radially to match the gain correction from the lens-shading correction block. A 33 node LUT is used to store 8-bit Sinter threshold offset values based on the square of the distance from the calculated centre. This allows a denser distribution of nodes closer to the edge of the image. Sinter threshold offsets are linearly interpolated for distances between LUT nodes.

The registers **Sinter: rm center x** and **Sinter: rm center y** specify the x and y centre coordinates and **Sinter: rm off center mult** specifies the scaling factor that can be calculated based on the furthest distance from the centre to the edge of the image. The register **Sinter: rm off center mult** is calculated as $2^{31}/r^2$ where r is the furthest distance, in pixels, from the centre to the edge of the image.

The following registers are used to configure this block.

Register	Description
rm enable	Set to 1 to enable radial modulation of Sinter.
rm center x	Center x coordinate used in radial Sinter.
rm center y	Center y coordinate used in radial Sinter.
rm off center mult	Scaling factor which maps the internal precision to the frame resolution. Based on the distance from the center to the furthest edge of the frame.
Sinter Shading: rm shading lut (0..32)	33 node Sinter radial shading table.

Note: *Please refer to the ISP_CONFIG_PING: sinter, ISP_CONFIG_PING: sinter shading, ISP_CONFIG_PONG: sinter, and ISP_CONFIG_PONG: sinter shading sections in the Mali-IV009 register map for the detailed register descriptions.*

2.7.14 Temper (temporal noise reduction)

This module is a motion-adaptive temporal noise reduction filter. It recursively averages the current frame with the history of previous frames, with a recursion level set locally by the degree of local motion detected in the current frame. The filter works in the raw domain and requires two frames of external memory. The data bit width is the video bit width + 4 bits. Note that in WDR mode the video bit width will be larger.

The Recursion_Limit can be increased or decreased to adjust the depth of recursion, which in turn sets the effective number of frames over which temporal averaging is performed, depending on the extent of motion in a given region. Increasing this parameter leads to a smaller recursion depth, with smaller noise reduction effect, and minimized motion artefacts.

- When Recursion_Limit is set to 0, up to 16 frames can be averaged.
- When Recursion_Limit is set to 0xf, frames are not averaged. This is equivalent to disabling temper.

The Temper thresholds are used to adjust Temper noise filter strength. Thresh_Long and Thresh_Short parameters corresponding to the long and short exposures respectively are used in WDR mode. When the exposure ratio is 1 or when WDR is disabled Thresh_Long and Thresh_Short should be set to the same value. The threshold values are determined during a standard calibration procedure using images captured at various ISO values and are modulated with system gain.

Larger values for Thresh_Short and Thresh_Long increase the strength of the noise reduction.

The performance of this module is governed by an externally-generated sensor Noise Profile LUT.

The reference data is stored in the DDR through DMAs. There are 2 DMA reader and 2 DMA writers to manage the reference data storage. 2 sets of DMAs are used only in Temper3 mode. When the temper operates in temper2 mode, only 1 DMA writer and 1 DMA reader is used.

2.7.14.1 Temper modes

2.7.14.1.1 Temper3 mode (16-bit data)

The following figure shows the Temper3 16 bit mode.

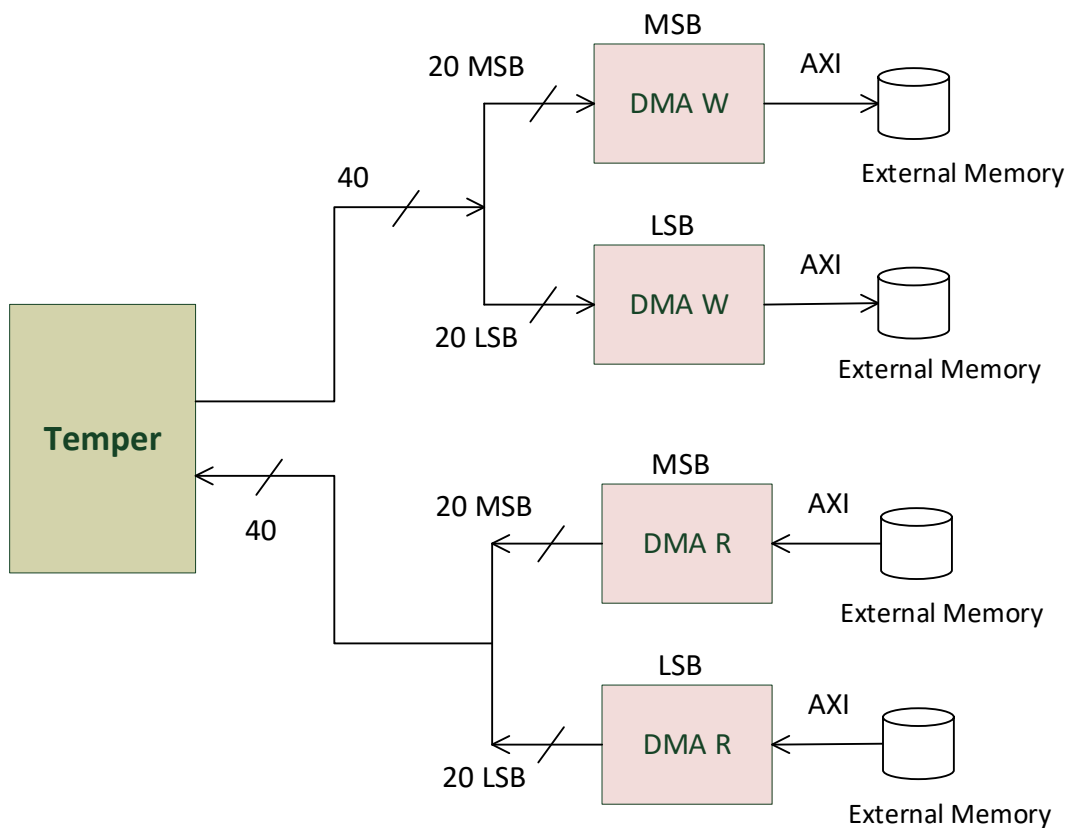


Figure 19. Temper3 16 bit mode

The temper module always provides the reference data in the following format.

Future reference		Previous reference	
4bits (Meta data)	16bits (Data)	4bits (Meta data)	16bits (Data)

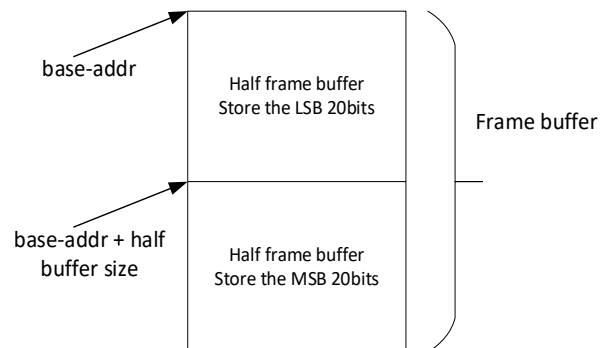
- Future reference is the next frame that will come up.

- Previous reference is the present frame being processed.

When the temper operates in 16-bit temper-3 mode, the 40-bit temper data is split into 2 equal halves. LSB 20-bits and MSB 20-bits are stored in two different buffers through 2 different DMAs called LSB-DMA and MSB-DMA respectively.

ISP_CONFIG:PING:temper dma: temper dw and ISP_CONFIG:PONG:temper dma: temper dw must be set to 0 (ZERO).

The Frame buffer can be assumed as 2 half frame buffers like the following diagram.



ISP_CONFIG_PING: temper dma:lsb bank base writer/reader and ISP_CONFIG_PONG: temper dma:lsb bank base writer/reader must have the base address of the whole frame buffer. The corresponding MSB dma reader and writer must have the base addresses greater than or equals to LSB dma base address + half buffer size.

2.7.14.1.2 Temper2 mode (16-bit data)

The following figure shows the Temper2 16 bit mode.

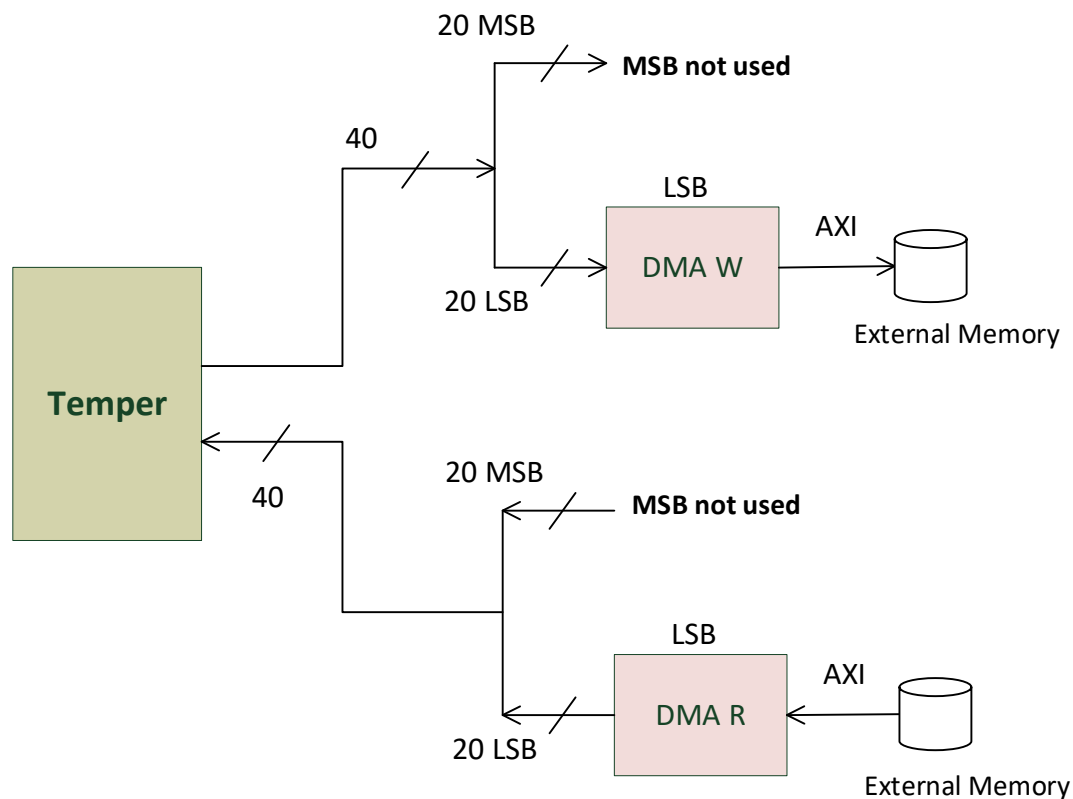


Figure 20. Temper2 16 bit mode

The MSB is switched off internally and is inactive.

The temper module always provides the reference data in the following format.

Future reference		Previous reference	
Xxxx	xxxx	4bits (Meta data)	16bits (Data)

When the temper operates in 16-bit temper 2 mode, the 40-bit temper data is split into 2 equal halves. LSB 20-bits are stored in a frame buffer buffers through the LSB DMA. The MSB 20-bits are ignored.

ISP_CONFIG:PING:temper dma: temper dw and ISP_CONFIG:PONG:temper dma: temper dw must be set to 0 (ZERO).

ISP_CONFIG_PING: temper: lsb bank base writer/reader and ISP_CONFIG_PONG: temper: lsb bank base writer/reader must have the base address of the frame buffer.

ISP_CONFIG_PING: temper dma:format and ISP_CONFIG_PONG: temper dma:format must be programmed with 20 (YUV 4:2:2 10bit packed).

2.7.14.1.3 Temper3 mode (12-bit data)

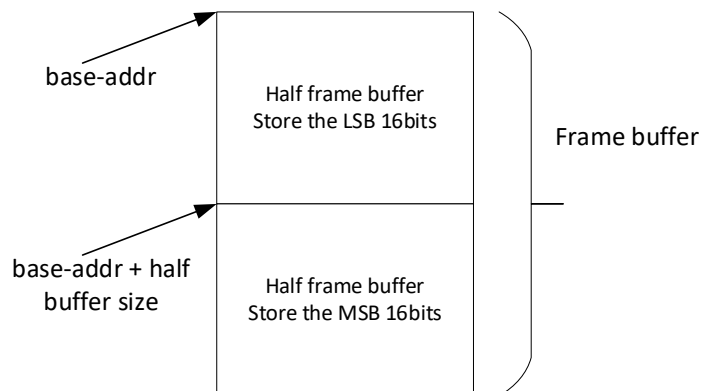
The temper module always provides the reference data in the following format.

Future reference			Previous reference		
4bits (Meta data)	12bits (Data)	0000	4bits (Meta data)	12bits (Data)	0000

When the temper operates in 12-bit temper 3 mode, the 40-bit temper data is split into 2 equal halves. LSB 16-bits [19:4] and MSB 16-bits [39:24] are stored in two different buffers through 2 different DMAs called LSB-DMA and MSB-DMA respectively.

ISP_CONFIG:PING:temper dma: temper dw and ISP_CONFIG:PONG:temper dma: temper dw must be set to 1 (ONE).

The Frame buffer can be assumed as 2 half frame buffers like the following diagram.



ISP_CONFIG_PING: temper dma:lsb bank base writer/reader and ISP_CONFIG_PONG: temper dma:lsb bank base writer/reader must have the base address of the whole frame buffer. The corresponding MSB dma reader and writer must have the base addresses greater than or equals to LSB dma base address + half buffer size.

ISP_CONFIG_PING: temper dma:format and ISP_CONFIG_PONG: temper dma:format must be programmed with 6 (RAW16).

2.7.14.1.4 Temper2 mode (12-bit data)

The temper module always provides the reference data in the following format.

Future reference			Previous reference		
Xxxx	Xxxx	xxxx	4bits (Meta data)	16bits (Data)	0000

When the temper operates in 12-bit temper 3 mode, the 40-bit temper data is split into 2 equal halves. LSB 16-bits [19:4] are stored in buffers through LSB DMA. The MSB data is ignored and the MSB DMA is unused.

ISP_CONFIG:PING:temper dma: temper dw and ISP_CONFIG:PONG:temper dma: temper dw must be set to 1 (ONE).

ISP_CONFIG_PING: temper dma:lsb bank base writer/reader and ISP_CONFIG_PONG: temper dma:lsb bank base writer/reader must have the base address of the whole frame buffer

ISP_CONFIG_PING: temper dma:format and ISP_CONFIG_PONG: temper dma:format must be programmed with 6 (RAW16)

2.7.14.2 Temper frame buffer prefetch

The temper module demands the reference frame when it is requested. To ensure that data availability, temper DMA needs to perform data prefetch. There are multiple options to perform the prefetch,

2.7.14.2.1 Multi-context case

In this case, the current context base address must be known before DMA can start the prefetch. Only when the 1st pixel comes in, its guaranteed that the configuration controller will swap to the correct configuration and then only the prefetch can be started. In this mode the read start happens automatically when the 1st input pixel comes in. The Sinter-delay line latency is used to balance the external memory read latency

Note: *The Sinter delay-line latency is used to ensure this delay. So the sinter can NOT be bypassed if the temper is enabled in multi-context mode. If the sinter functionality is not needed, but still temper is expected to be enabled, please disable sinter through ISP_CONFIG_PING:Sinter:Enable and ISP_CONFIG_PONG:Sinter:Enable signal. In the single context mode, the Sinter can also be bypassed if the ISP_COMMON:isp global chicken bit:rd start sel signal is set to 1.*

2.7.14.2.2 Single context mode

In the single context mode, the default prefetch start mode is same as in multi-context mode. Its recommended to use the same flow.

Note: *Please refer to the **ISP_CONFIG_PING: temper/temper dma** and **ISP_CONFIG_PONG: temper/temper dma** sections in the **Mali-IV009 register map** for the detailed register descriptions.*

2.7.15 Chromatic Aberration Correction (CAC)

CAC uses a 13 x 13 kernel to correct longitudinal chromatic aberration effects across the image. The image is segmented into 127 x 127 regions. Calibration data defines the correction transform for each region.

The CAC module may be configured for correction of three-channel and four-channel data, including support for IR channel correction.

The correction factor per region is represented by a pseudo-floating point value, able to represent shifts of as many as +/- 26 native pixels and a maximum resolution of 1/256 pixel.

The following register is used to enable this block.

Register	Description
CA Correction: Enable	Chromatic Aberration Correction enable: 0 = off, 1 = on.

Note: *Please refer to the **CA_CORRECTION_MESH_PING_MEM** and **CA_CORRECTION_MESH_PONG_MEM** sections in the **Mali-IV009 register map**.*

2.7.16 Square

The square module converts the 16-bit square root data back to 20-bit linear data. The black level is subtracted in the square root domain and then added back in the linear domain

2.7.17 Black Level Correction

This module provides removal of the sensor black level from the sensor data. Separate offsets are provided for each of the 4 Bayer component color channels (R, Gr, Gb, B).

The following registers are used to configure this block.

Register		Description
Sensor Offset pre shading	offset 00	Black offset subtraction for each channel in linear domain: Channel 00 (R).
	offset 01	Black offset subtraction for each channel in linear domain: Channel 01 (Gr).
	offset 10	Black offset subtraction for each channel in linear domain: Channel 10 (Gb).
	offset 11	Black offset subtraction for each channel in linear domain: Channel 11 (B).

Note: *Please refer to the **ISP_CONFIG_PING: sensor offset pre shading**, and **ISP_CONFIG_PONG: sensor offset pre shading** sections in the Mali-IV009 register map for the detailed register descriptions.*

2.7.18 Lens shading correction

The ISP features both radial and mesh lens shading correction. Radial shading correction exploits the radial nature of lens shading and can correct off-centre and oval shading effects. Mesh shading correction provides a further correction for non-linear shading distortions and fine-tunes the effect produced by radial shading correction. Mesh shading correction is applied to the image using a grid of 64 x 64 zones.

2.7.18.1 Radial shading correction

This module performs vignetting (radial shading) correction.

Shading correction is applied per color layer, supporting a 4th channel for IR, and multiple concurrent settings are stored to consider different and mixed lighting color temperature.

During radial shading correction, the lens shading is corrected through a radial model in which the position of the shading function for each color channel can be independently set. Radial shading correction coefficients are provided through independent 33-element LUTs. This assumes symmetric lens shading based on the distance from the calculated centre. The node positions are based on the square of the distance from the centre which allows a denser distribution of nodes closer to the edge of the image.

The registers are described in the table below. The values and the LUT table values can all be obtained from the ARM ISP Calibration tool using flat field images.

The radial shading coefficients are stored in a 32-bit, 4 x 129-entry LUT. Coefficients are in x.12 format where the lower 12-bits are fractional values. The coefficients are stored starting from the centre to the outer edge for each color plane. Red coefficients are stored in locations 0 to 129, green coefficients are stored in locations 256 to 384, blue coefficients are stored in locations 512 to 640 and the Ir channel coefficients are stored in locations 768 to 896.

There are bubbles between the coefficient sets. Any attempt to any write in the bubble locations is rejected.

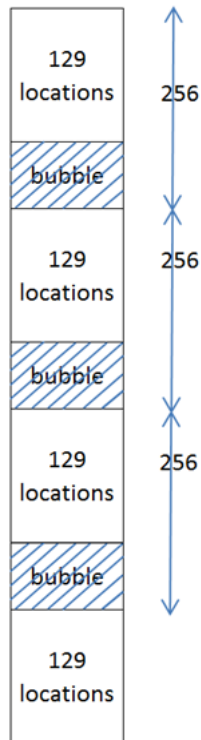


Figure 21. Radial shading memory layout

The following registers are used to configure this block.

Register	Description
Enable	Set to 1 to enable radial shading correction.
centerR x, centerG x, centerB x, centerIr x	Center x coordinates for R, G, B and IR shading map.
centerR y, centerG y, centerB y, centerIR y	Center y coordinates for R, G, B and IR shading map.
off center multRx, off center multGx, off center multBx, off center multIRx	Normalizing X factor which scales the R, G, B or IR radial table to the edge of the image. Calculated as $2^{31}/R^2$ where R is the furthest distance from the center coordinate to the edge of the image in pixels.

Register	Description
off center multRy, off center multGy, off center multBy, off center multIRy	Normalizing Y factor which scales the R, G, B or IR radial table to the edge of the image. Calculated as $2^{31}/R^2$ where R is the furthest distance from the center coordinate to the edge of the image in pixels.
Radial Shading Mem(0..896)	4 x 129 node radial shading LUT for each color plane. Each node takes one 32-bit entry in the LUT with R starting address at base address + 0, G starting address at base address + 256, B starting address at base address + 2*256 and Ir starting from base address + 3*256.

Note: *Please refer to the **ISP_CONFIG_PING: radial shading** and **ISP_CONFIG_PONG: radial shading** sections in the Mali-IV009 register map for the detailed register descriptions.*

2.7.18.2 Mesh shading correction

Mesh shading provides a further correction for non-linear shading distortions. This is applied to the image according to 64 x 64 zones, again considering color plane and lighting color temperature.

The ISP provides four pages (R, G, B and IR) each containing an intensity correction table. Each table provides a corrective gain for each pixel in the image by using bi-linear interpolation from a mesh of points. A maximum of 64 x 64 points for each color plane can be used. For each point in the mesh there is a single 8-bit coefficient representing the intensity gain value. Mesh points, **Mesh(row,col)**, are numbered in raster order with **Mesh(0,0)** representing the top left corner of the image.

All of the possible 8-bit coefficients are stored in a single 32-bit LUT. Data is stored as a horizontal array of values defining the first line for the first color plane. Data continues line by line, then color plane by color plane. The first 1024 entries define the first page and color plane, entries 1025 to 2048 are for the second page and color plane, and entries 2049 to 3072 are for the last page and color plane. Each 32-bit LUT entry consists of four 8-bit mesh points.

For each of the three color planes/pages, as many as four tables can be stored to allow switching between mesh tables under different light source illuminants. The register **Mesh Shading: Mesh alpha mode** can be used to select the number of light source illuminants. Note the mapping of the mesh table changes according to the number of light sources used. This reduces the number of mesh points available for shading correction, but allows more shading tables under different light sources to be used.

Mesh alpha mode	Format of Mesh Table
0	One light source with a maximum of 64x64 nodes (64 rows by 64 columns) for each color plane.
1	Two light sources with a maximum of 32x64 nodes (32 columns by 64 rows) for each color plane and light source.
2	Up to Four light sources with a maximum of 32x32 nodes (32 rows by 32 columns) for each color plane and light source.
3	Two light sources with a maximum of 64x32 nodes (64 columns by 32 rows) for each color plane and light source.

Figure 22. Mesh table formats based on mesh alpha mode.

The following registers are used to configure this mode. These registers are all members of the **Mesh Shading** group:

Register	Description
Enable	Lens mesh shading correction enable: 0=off, 1=on.
Mesh show	Lens mesh shading correction debug: 0=off, 1=on (show mesh data).
Mesh scale	Selects the precision and maximal gain range of mesh shading correction Gain range: 00 -> 0..2; 01 -> 0..4; 02 -> 0..8; 03 -> 0..16; 04 -> 1..2; 05 -> 1..3; 06 -> 1..5; 07 -> 1..9 (float).
Mesh alpha mode	Sets alpha blending between mesh shading tables. 0 = no alpha blending; 1 = 2 banks (odd/even bytes) 2 = 4 banks (one per 8-bit lane in each dword).
Mesh strength	Mesh strength in 4.12 format, e.g. 0 = no correction, 4096 = correction to match mesh data. Can be used to reduce shading correction based on AE.

Note: *Please refer to the **ISP_CONFIG_PING: mesh shading and ISP_CONFIG_PONG: mesh shading** sections in the Mali-IV009 register map for the detailed register descriptions.*

The following tables show the configuration of the Mesh LUTs based on the **Mesh_alpha_mode** setting and the maximum number of nodes.

LUT Entry	0				1				...	1023			
Page Number	0				0				...	0			
Mesh point	0,3	0,2	0,1	0,0	0,7	0,6	0,5	0,4	...	63,63	63,62	63,61	63,60
Bits	24:31	16:23	8:15	0:7	24:31	16:23	8:15	0:7	...	24:31	16:23	8:15	0:7
Light Source	LS0	LS0	LS0	LS0	LS0	LS0	LS0	LS0	...	LS0	LS0	LS0	LS0
LUT Entry	1024				1025				...	2047			
Page Number	1				1				...	1			
Mesh point	0,3	0,2	0,1	0,0	0,7	0,6	0,5	0,4	...	63,63	63,62	63,61	63,60
Bits	24:31	16:23	8:15	0:7	24:31	16:23	8:15	0:7	...	24:31	16:23	8:15	0:7
Light Source	LS0	LS0	LS0	LS0	LS0	LS0	LS0	LS0	...	LS0	LS0	LS0	LS0
LUT Entry	2048				2049				...	3071			
Page Number	2				2				...	2			
Mesh point	0,3	0,2	0,1	0,0	0,7	0,6	0,5	0,4	...	63,63	63,62	63,61	63,60
Bits	24:31	16:23	8:15	0:7	24:31	16:23	8:15	0:7	...	24:31	16:23	8:15	0:7
Light Source	LS0	LS0	LS0	LS0	LS0	LS0	LS0	LS0	...	LS0	LS0	LS0	LS0
LUT Entry	2072				2073				...	4096			
Page Number	3				3				...	3			
Mesh point	0,3	0,2	0,1	0,0	0,7	0,6	0,5	0,4	...	63,63	63,62	63,61	63,60
Bits	24:31	16:23	8:15	0:7	24:31	16:23	8:15	0:7	...	24:31	16:23	8:15	0:7
Light Source	LS0	LS0	LS0	LS0	LS0	LS0	LS0	LS0	...	LS0	LS0	LS0	LS0

Figure 23. Mesh Shading LUT Configuration, Mesh Alpha Mode = 0, Mesh Correction for 1 Light source.

LUT Entry	0				1				...	1023			
Page Number	0				0				...	0			
Mesh point	0,1		0,0		0,3		0,2		...	63,31		63.30	
Bits	24:31	16:23	8:15	0:7	24:31	16:23	8:15	0:7	...	24:31	16:23	8:15	0:7
Light Source	LS1	LS0	LS1	LS0	LS1	LS0	LS1	LS0	...	LS1	LS0	LS1	LS0
LUT Entry	1024				1025				...	2047			
Page Number	1				1				...	1			
Mesh point	0,1		0,0		0,3		0,2		...	63,31		63.30	
Bits	24:31	16:23	8:15	0:7	24:31	16:23	8:15	0:7	...	24:31	16:23	8:15	0:7
Light Source	LS1	LS0	LS1	LS0	LS1	LS0	LS1	LS0	...	LS1	LS0	LS1	LS0
LUT Entry	2048				2049				...	3071			
Page Number	2				2				...	2			
Mesh point	0,1		0,0		0,3		0,2		...	63,31		63.30	
Bits	24:31	16:23	8:15	0:7	24:31	16:23	8:15	0:7	...	24:31	16:23	8:15	0:7
Light Source	LS1	LS0	LS1	LS0	LS1	LS0	LS1	LS0	...	LS1	LS0	LS1	LS0
LUT Entry	3072				3073				...	4096			
Page Number	3				3				...	3			
Mesh point	0,1		0,0		0,3		0,2		...	63,31		63.30	
Bits	24:31	16:23	8:15	0:7	24:31	16:23	8:15	0:7	...	24:31	16:23	8:15	0:7
Light Source	LS1	LS0	LS1	LS0	LS1	LS0	LS1	LS0	...	LS1	LS0	LS1	LS0

Figure 24. Mesh Shading LUT Configuration, Mesh Alpha Mode=1, Mesh Correction for 2 Light sources

LUT Entry	0				1				...	1023			
Page Number	0				0				...	0			
Mesh point	0,0				0,1				...	31,31			
Bits	24:31	16:23	8:15	0:7	24:31	16:23	8:15	0:7	...	24:31	16:23	8:15	0:7
Light Source	LS3	LS2	LS1	LS0	LS3	LS2	LS1	LS0	...	LS3	LS2	LS1	LS0
LUT Entry	1024				1025				...	2047			
Page Number	1				1				...	1			
Mesh point	0,0				0,1				...	31,31			
Bits	24:31	16:23	8:15	0:7	24:31	16:23	8:15	0:7	...	24:31	16:23	8:15	0:7
Light Source	LS3	LS2	LS1	LS0	LS3	LS2	LS1	LS0	...	LS3	LS2	LS1	LS0
LUT Entry	2048				2049				...	3071			
Page Number	2				2				...	2			
Mesh point	0,0				0,1				...	31,31			
Bits	24:31	16:23	8:15	0:7	24:31	16:23	8:15	0:7	...	24:31	16:23	8:15	0:7
Light Source	LS3	LS2	LS1	LS0	LS3	LS2	LS1	LS0	...	LS3	LS2	LS1	LS0
LUT Entry	3072				3073				...	4096			
Page Number	3				3				...	3			
Mesh point	0,0				0,1				...	31,31			
Bits	24:31	16:23	8:15	0:7	24:31	16:23	8:15	0:7	...	24:31	16:23	8:15	0:7
Light Source	LS3	LS2	LS1	LS0	LS3	LS2	LS1	LS0	...	LS3	LS2	LS1	LS0

Figure 25. Mesh Shading LUT Configuration, Mesh Alpha Mode=2, 4 Light sources

LUT Entry	0				1				...	1023			
Page Number	0				0				...	0			
Mesh point	0,1		0,0		0,3		0,2		...	31,63		31,62	
Bits	24:31	16:23	8:15	0:7	24:31	16:23	8:15	0:7	...	24:31	16:23	8:15	0:7
Light Source	LS1	LS0	LS1	LS0	LS1	LS0	LS1	LS0	...	LS1	LS0	LS1	LS0
LUT Entry	1024				1025				...	2047			
Page Number	1				1				...	1			
Mesh point	0,1		0,0		0,3		0,2		...	31,63		31,62	
Bits	24:31	16:23	8:15	0:7	24:31	16:23	8:15	0:7	...	24:31	16:23	8:15	0:7
Light Source	LS1	LS0	LS1	LS0	LS1	LS0	LS1	LS0	...	LS1	LS0	LS1	LS0
LUT Entry	2048				2049				...	3071			
Page Number	2				2				...	2			
Mesh point	0,1		0,0		0,3		0,2		...	31,63		31,62	
Bits	24:31	16:23	8:15	0:7	24:31	16:23	8:15	0:7	...	24:31	16:23	8:15	0:7
Light Source	LS1	LS0	LS1	LS0	LS1	LS0	LS1	LS0	...	LS1	LS0	LS1	LS0
LUT Entry	3072				3073				...	4096			
Page Number	3				3				...	3			
Mesh point	0,1		0,0		0,3		0,2		...	31,63		31,62	
Bits	24:31	16:23	8:15	0:7	24:31	16:23	8:15	0:7	...	24:31	16:23	8:15	0:7
Light Source	LS1	LS0	LS1	LS0	LS1	LS0	LS1	LS0	...	LS1	LS0	LS1	LS0

Figure 26. Mesh Shading LUT Configuration, Mesh Alpha Mode=3, Mesh Correction for 2 Light sources

2.7.19 Static white balance

Refer to section [2.7.32.1](#) Auto white balance for more information.

Note: *Please refer to the **ISP_CONFIG_PING: white balance and ISP_CONFIG_PONG: white balance** sections in the Mali-IV009 register map for the detailed register descriptions.*

2.7.20 Precision management

The precision management function maps 20-bits of linear dynamic range to 12-bits. The ISP provides multiple approaches to this process.

2.7.20.1 Display functions

Iridix is a local tone-mapping engine, but it can be used for global tone-mapping, using LUTs of input and output values.

In both cases, first set `black_level_amp0` to 1. This indicates that the input images are already corrected for black level (sensor offset).

Using Iridix for local tone-mapping

1. Set `iridx.gtm_select` to 0. This enables local tone-mapping.
2. Follow the tuning procedures provided in the ISP Tuning Guide.

Using Iridix for global tone-mapping

1. Define the global tone curve by setting `iridx_lut.globaltm_x_lut` (33 input values of 24-bit precision) and `iridx_lut.globaltm_y_lut` (33 output values of 24-bit precision).
2. Set `iridx.gtm_select` to 1. This disables local tone-mapping and enables global tone-mapping using the LUT you have defined.

For display purposes Iridix provides local tone mapping, ensuring maximum use of the dynamic range at the output. This is particularly powerful in revealing the details in dark or shadow areas of the image.

The Iridix module is an advanced local tone mapping engine using space-variant dynamic-range compression. The technology is based on the performance of the human visual system. It adjusts the tonal range of the input image so that it is optimized for viewing on standard output devices.

The primary function of this module is to transfer up to 16-bit WDR data into standard 10-bit output suitable for post-processing by the latter part of the pipeline.

Note that the asymmetry LUT is applicable to only Iridix tone mapping, not the global tone mapping options described elsewhere.

Note that the block "Intensity Mapping" in Figure 26 is a hard-coded fixed function, used to map raw color to intensity, and is not adjustable by the customer. It is shown for reference only.

The following registers are used to configure this module. These registers are all members of the **Iridix** group.

Register	Description
Iridix_on	Enable/disable Iridix. With this set to 0 (off), local tone mapping will not be applied to the output image. This means the image input and the output of the Iridix processing block will be identical, regardless of other Iridix parameter values.
Variance_space	Degree of spatial sensitivity of the <i>Dynamic Range Compression</i> (DRC) algorithm. This is an algorithm used internally by the Iridix module.
Variance_intensity	Degree of intensity sensitivity of the DRC algorithm.
Slope_max	Restricts the maximum slope (gain) which can be generated by the adaptive algorithm.
Slope_min	Restricts the minimum slope (gain) which can be generated by the adaptive algorithm.
Black_level	Iridix black level. Values below this are not processed and remain linear.
White_level	Iridix white level. Values above this are not processed and remain linear.

Note: *The use of Iridix and global tone mapping are mutually exclusive and may not be used together.*

Please refer to the **ISP_CONFIG_PING: iridix** and **ISP_CONFIG_PONG: iridix** sections in the Mali-IV009 register map for the detailed register descriptions.

2.7.20.2 Global tone mapping

Global tone mapping is a 2d look up table. The data is compared against values in the **X_lut** and result of this is used to select a pair of values from the **Y_lut**. The result is obtained by linear interpolation between the Y values.

Where a data-invariant compression approach is required, multiple options are provided:

- Gamma DL compression function.
- Look-up-table (LUT) providing a PWL curve.
- Square-root function.
- Non-equidistant LUT (allowing the available codes to be distributed efficiently throughout the curve).

Note: *The `iridx.black_level_amp0` function is used for both Global Tone Mapping and Iridix.*

2.7.21 East-west mirror

IP provides an east-west switch module so that the image can be horizontally flipped before processed image is sent out. This module is placed after iridx to minimize the SRAM storage requirement. When the mirroring is enabled, all the modules before Mirror and after mirror must have appropriate RGGB pattern start point indicator. The following registers must be programmed correctly when mirror is enabled.

ISP_CONFIG_PING:top: rggb start pre mirror

ISP_CONFIG_PONG:top: rggb start pre mirror

ISP_CONFIG_PING:top: rggb start post mirror

ISP_CONFIG_PONG:top: rggb start post mirror

2.7.22 Multi-CFA demosaic

2.7.22.1 RGGB Demosaic

The Demosaic unit is responsible for reconstructing a full color image from the incomplete (spatially undersampled) color samples output by an image sensor overlaid with a Color Filter Array (CFA). Additionally, this module provides advanced control over image sharpening.

Digital camera sensor elements (pixel sensors) by themselves, can record only the intensity of light falling on them and not differentiate between different colors, hence producing only a grayscale image. To capture color information, filters must be placed over each pixel sensor, that permit only particular colors of light through. The filters used must allow reconstruction of a full color image with Red, Green and Blue (RGB) values for each pixel location. The most common type of such Color Filter Array is called a "Bayer array," shown in Figure 28, so called because the color filters are arranged in a regular pattern with each 2x2 pixel group having an RGGB arrangement.

- Half of all the pixels are green (G), while a quarter each are red (R) and blue (B).
- Green cells in the same row as blue are labeled Gb and the green cells in the same row as red are labelled Gr.
- The pattern can start with any of R, Gr, Gb or B.

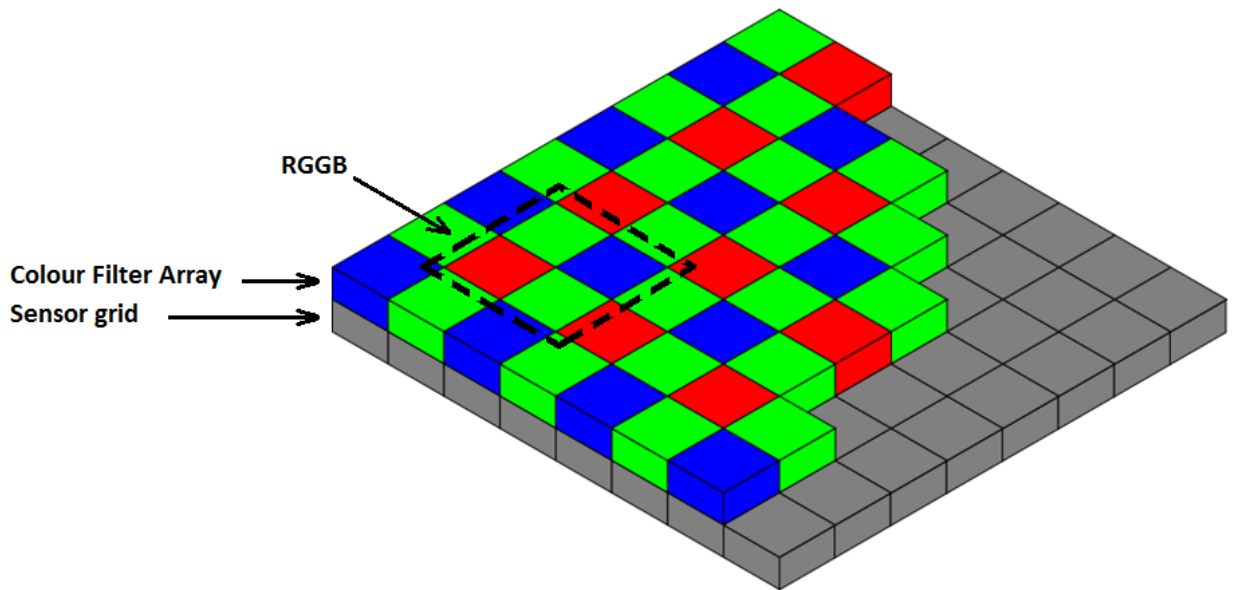
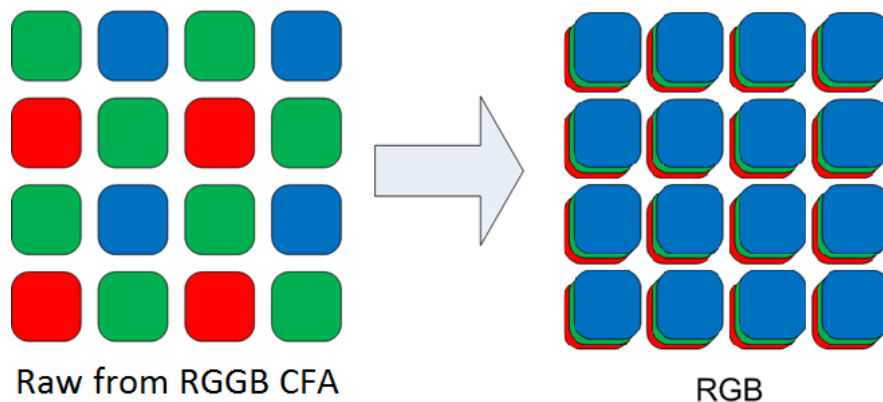


Figure 27. Bayer array

This arrangement of color filters essentially causes spatial undersampling of color information. The demosaic unit is responsible for reconstructing a full color image (RGB for each pixel) from this incomplete color information as shown below:



The module consists of many filters, which reconstruct chrominance channels based on interpolated luminance channel. It also considers signal-dependent sensor noise (based on earlier determined noise profile) to preserve the sharpness of the edges and smoothness of uniform areas, while interpolating the missing pixel components. Hence, the interpolation of missing pixel components explicitly incorporates the noise characteristics of the sensor. The embedded sharpening also minimizes the amplification of high frequency noise.

2.7.22.2 RGBIr Demosaic

This demosaic block converts a single plane with a 2 x 2 RGBIR CFA pattern to four dense planes for R, G, B and IR. The resulting output planes are full resolution at the native resolution of the sensor, meaning a full resolution IR plane is available at the output. The process takes place in the following main stages.

2.7.22.2.1 Remove blooming from IR channel

Significant blooming of the IR channel can occur at the borders of regions where the IR channel is clipped. If this blooming is not corrected, IR correction will result in unwanted black artefacts in these locations.

IR pixels which have values above the setting of **Demosaic RGBIR: Clip Debloom** and are near clipped IR pixels, are deemed to be the result of blooming, and are corrected. This should usually be equal to, or slightly below, **Demosaic RGBIR: Clip Level**.

2.7.22.2.2 Highlight recovery (declipping) and color correction

If, in a region of the image, one or more of the three visible channels is clipped, then IR correction will fail, resulting in unnatural colors. The purpose of declip is to correct these regions before IR correction is applied. Declip will modify only the R, G, and B channels.

Pixel values above the value of **Demosaic RGBIR: Clip Level** are deemed to be clipped, and will be corrected by the declip algorithm.

Demosaic RGBIR: Declip Mode: In regions where all three visible channels (R, G and B) are clipped, the declip algorithm has no local information to use to fix colors. In such regions there are two possible fallback behaviors:

- **Demosaic RGBIR: Declip Mode = 0:** The algorithm will replace clipped content using the grey level of the row above. For small regions this looks acceptable, but for larger regions it can result in diagonal grey streaks. If the illuminant has high IR content, then this mode will probably give the best result.
- **Demosaic RGBIR: Declip Mode = 1:** The algorithm will replace clipped content with a high constant value (that is, white). If the illuminant has relatively low IR content, then this is mode will probably give the best result. If this mode is used when IR level is high, the resulting white areas may appear too bright compared to their surroundings (this has been termed *snow*).

This parameter has no effect in regions where one or more of the visible channels is not clipped. Ideally, exposure should be set to ensure that at least one of the visible channels is everywhere unclipped.

2.7.22.2.3 White Balance

White balance is applied at two points in the algorithm. It is first applied before interpolation, in order to balance the channels and improve the quality of interpolation. This

we refer to as static white balance. The process uses the parameters **Demosaic RGBIR: Static Gain R**, **Demosaic RGBIR: Static Gain B**, and **Demosaic RGBIR: Static Gain I**. After interpolation this static white balance is inverted, IR correction is applied and then the final white balance is applied. The final white balance uses parameters **Demosaic RGBIR: Gain R** and **Demosaic RGBIR: Gain B**.

Demosaic RGBIR: Static Gain R, **Demosaic RGBIR: Static Gain B**, and **Demosaic RGBIR: Static Gain I**: These should reflect the relative levels of the four channels before IR correction. They need not be completely accurate, as they will be inverted after interpolation. If accurate values can be obtained, they can be used to compute the two final white balance gains using the IR correction matrix.

Demosaic RGBIR: Gain R and **Demosaic RGBIR: Gain B**: These should reflect the relative levels of three visible channels (R, G and B) after IR correction. These will not be the same as **Demosaic RGBIR: Static Gain R**, and **Demosaic RGBIR: Static Gain B**. These are the last white balance values to be applied. If it is not possible to gather image statistics after IR correction, then these two gains can be computed from the three static gains using the IR correction matrix.

2.7.22.2.4 Interpolation

Demosaic RGBIR: Interpolation Directionality: A lower value means the effect will be more directional. Larger values will result in a more bilinear interpolation. This should be modulated according to the expected noise level.

2.7.22.2.5 Sharpening

Sharpening will be applied to all output planes. The degree of sharpening is determined by the following parameters:

Demosaic RGBIR: Sharp Low: The strength of sharpening applied in regions deemed to be flat. Which regions are deemed to be flat or directional will be determined according to **Demosaic RGBIR: Interpolation Directionality**.

Demosaic RGBIR: Sharp High: The strength of sharpening applied in regions deemed to be directional. Which regions are deemed to be flat or directional will be determined according to **Demosaic RGBIR: Interpolation Directionality**.

Demosaic RGBIR: Sharp Limit: The maximum amplitude of sharpening. Use this register to prevent overshoot on very high contrast edges.

2.7.22.2.6 IR correction

This process recovers natural-looking colors from IR-corrupted R, G and B planes caused by the absence of an IR-cut filter, and removes visible light corruption of the IR plane if required.

Demosaic RGBIR: IR Correct Mat XX: These 16 values are the coefficients of the 4 x 4 matrix used for IR correction. The order of the rows, and columns, is R, G, B, IR. This matrix can be computed from quantum efficiency curves of the sensor used. Once computed, it should not need to change unless a different sensor is used.

The following registers are used to configure this mode. The registers in the following table are all found in the **Demosaic RGBIR** group in the register map with the exception of the first parameter, **RGGB start**. This register, from the **Top** group, is used elsewhere in the ISP but in RGBIR processing is used to specify the position of the R cell in the four-pixel group. Permitted ranges and default register values for 12-bit, 14-bit, and 16-bit data can be found in the **Register Map**.

Register	Description
Top: RGGB start	<div> <div> <div>0</div> <div>1</div> <div>2</div> <div>3</div> </div> <div>Position of the R cell in the four pixel group (2-bit integer).</div> </div>
Clip Debloom	Threshold used for deblooming. Will usually be below Clip Level .
Clip Level	Pixel values above this are deemed to be clipped.
Declip Mode	Method of declipping where R, G, and B are all clipped: 0: Use local intensity to estimate. 1: Use constant value of 1.
FC Low	Value to start color aliasing alpha ramp.
FC Grad	Gradient of color aliasing alpha ramp.
Gain R	White balance value for IR corrected R channel.
Gain B	White balance value for IR corrected B channel.
Interpolation Directionality	Sensitivity of directional interpolation to local gradient. A lower value means more sensitive. Increasing the value will eventually result in bilinear demosaic.
IR Correct MatXX	A 4 x 4 matrix of values for correcting the corruption of visible (R, G, B) channels by IR. Can also be used to correct influence of visible light on IR channel.

Register	Description
RGBIr config	Selects the output information for debugging: 0: Standard output (default). 1: clipMap - showing which pixels are clipped. 2: fcAlpha - the alpha map used for false-color/color-aliasing correction. 3: directional - showing where directional interpolation has been applied. 4: Gg - the grey used to determine edge direction for interpolation. 5: RGB after static WB, but before IR correction or final WB. 6: Grey - the grey used to desaturate, for false color correction.
Sharp High	Sharpening strength in edge and detail regions.
Sharp Limit	Limit sharpening amount to prevent overshoot.
Sharp Low	Sharpening strength in flat regions.
Static Gain B	White balance value for non IR corrected B channel.
Static Gain I	White balance value for non IR corrected IR channel.
Static Gain R	White balance value for non IR corrected R channel.

2.7.23 Purple fringing correction

Purple Fringing is a form of chromatic aberration which causes an undesirable purple or blue hue around high contrast edges. The Purple Fringing Correction module removes these by isolating the affected areas using masks for hue, saturation, luma, edge detection and radial position. The fringing is then desaturated.

Note: *Please refer to the ISP_CONFIG_PING: pf correction and ISP_CONFIG_PONG: pf correction sections in the Mali-IV009 register map for the detailed register descriptions.*

2.7.24 Color correction matrix (4x3)

This module applies linear color correction to input {R, G, B} or {R, G, B, Ir} pixel values. The Matrix coefficients are usually calculated dynamically by the Firmware in order to implement color saturation adjustments and color-correction under various lighting conditions which have been calibrated for the sensor.

The 4 x 3 calculation for the RGBIr CFA pattern is as follows:

$$\begin{pmatrix} Out1 \\ Out2 \\ Out3 \end{pmatrix} = \begin{pmatrix} A11 & A12 & A13 & A14 \\ A21 & A22 & A23 & A24 \\ A31 & A32 & A33 & A34 \end{pmatrix} \times \begin{pmatrix} In1 \\ In2 \\ In3 \\ In4 \end{pmatrix}$$

In1, In2, In3 and In4 are the inputs (R, G, B and Ir respectively) and A11 to A34 are configurable matrix coefficients. The coefficients are stored as 13 bit values in s4.8 number format, where the MSbit is used as a sign bit. Negative values have the MSbit set to 1.

If the CFA pattern is RGGB, then the Ir coefficients and the Ir channel offset must be set to 0 (ZERO).

Where the variable names in the formula, A11, A12 and so on, are taken from the following registers in the **Output formatter** group:

Variable	Register	Function	Format
RGB to RGB 4 x 3 matrix			
A11	ccm:Coefft R-R	R-R multiplier for RGB pipeline.	Sign/magnitude 4.8-bit fixed-point. The MSB is sign.
A12	ccm:Coefft G-R	G-R multiplier for RGB pipeline.	Sign/magnitude 4.8-bit fixed-point. The MSB is sign.
A13	ccm:Coefft B-R	B-R multiplier for RGB pipeline.	Sign/magnitude 4.8-bit fixed-point. The MSB is sign.
A14	ccm:Coefft IR-R	IR-R multiplier for RGB pipeline.	Sign/magnitude 4.8-bit fixed-point. The MSB is sign.
A21	ccm:Coefft R-G	R-G multiplier for RGB pipeline.	Sign/magnitude 4.8-bit fixed-point. The MSB is sign.
A22	ccm:Coefft G-G	G-G multiplier for RGB pipeline.	Sign/magnitude 4.8-bit fixed-point. The MSB is sign.
A23	ccm:Coefft B-G	B-G multiplier for RGB pipeline.	Sign/magnitude 4.8-bit fixed-point. The MSB is sign.
A24	ccm:Coefft IR-G	IR-G multiplier for RGB pipeline.	Sign/magnitude 4.8-bit fixed-point. The MSB is sign.
A31	ccm:Coefft R-B	R-B multiplier for RGB pipeline.	Sign/magnitude 4.8-bit fixed-point. The MSB is sign.
A32	ccm:Coefft G-B	G-B multiplier for RGB pipeline.	Sign/magnitude 4.8-bit fixed-point. The MSB is sign.
A33	ccm:Coefft B-B	B-B multiplier for RGB pipeline.	Sign/magnitude 4.8-bit fixed-point. The MSB is sign.
A34	ccm:Coefft IR-B	IR-B multiplier for RGB pipeline.	Sign/magnitude 4.8-bit fixed-point. The MSB is sign.

Note: *Please refer to the **ISP_CONFIG_PING: ccm** and **ISP_CONFIG_PONG: ccm** sections in the Mali-IV009 register map for the detailed register descriptions.*

2.7.25 Color noise reduction (CNR)

This block reduces the Chroma Noise in the image by using wider Gaussian kernels to target the noise that typically affects the larger areas of the image. During the process the block keeps the intensity content of the image intact and only process the Chroma part of it.

The block converts the RGB image into YUV domain and process the U and V channels individually by further segmenting it based on color content. The Gaussian kernels are then applied to each U and V channels of the corresponding segment and are configured by corresponding offset or slope parameters. To minimize the hardware implementation or reduce the number of lines for large kernels, the vertical Gaussian filtering is replaced by a novel recursive filter. Also, to reduce the area further, the color channels are down-sampled before the processing and then up-sampled at the output stage. The output U and V channels are blend of the processed U and V and original U and V channels respectively which is configured by delta offset or slope parameters. The processed U and V and unprocessed Y are converted back to the RGB domain.

2.7.26 Non-equidistant gamma and 3D LUT

Most of the third-party 3D lut coefficients are in the sRGB domain. So, a 65-node non-equidistant gamma is used to convert the data to the sRGB domain.

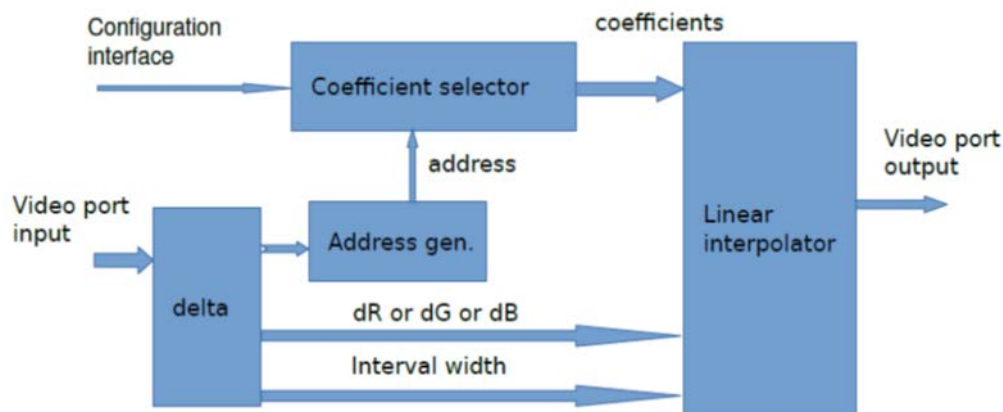


Figure 28. 1-channel sRGB conversion

Figure 28 shows a 1-channel sRGB conversion. Three of such instances are used for RGB data conversion. No equidistant LUT consists of 8 segments. Each segment is a 1D lookup table with equidistant node distribution.

The following table shows the x-position in the gamma curve.

Pixel in Range	Sector	No. of segment	Segment width
0-31	0	16	2
32-62	1	4	8

Pixel in Range	Sector	No. of segment	Segment width
64-127	2	4	16
128-255	3	4	32
256-511	4	4	54
512-1023	5	4	128
1024-2047	6	4	256
2048-8191	7	24	256

The 3D LUT module realizes three independent nonlinear processing of three input color plans. The top level of the module consists of three instances of three dimensional look up table with three input color planes and one color plane output. All LUT_3D_3TO1 receive the same streamed input video and configuration data, read coefficients from different memory units (SRAM), and realize bi-linear interpolation.

The present realization of LUT_3D has the following new features:

- The independent size of LUT for each input color plane.
- The number of LUTs nodes can be any positive integer value.
- The LUTs coefficients value can be as positive as negative.
- the pre interpolation stage to recalculate the values of coefficients of LUTs for increasing the quality of the interpolation.

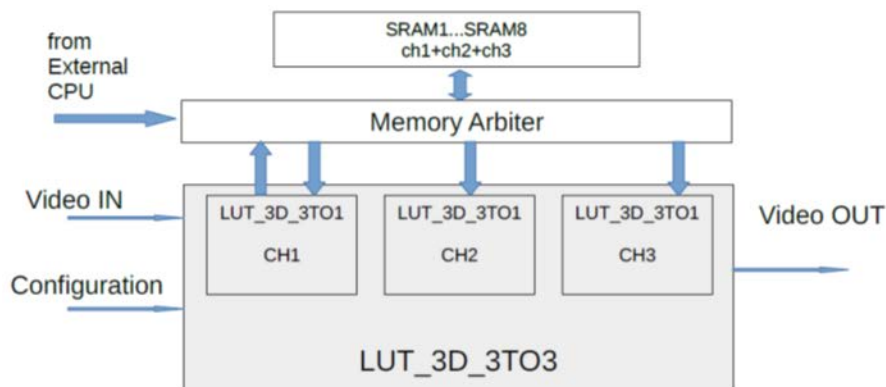


Figure 29. LUR 3D

2.7.27 Crop

This module enables you to select an arbitrary area of the image to form the final output image. The resolution of the output is that of the cropped area, no scaling is applied.

Note: *Cropping results in increased horizontal and vertical blanking at the output frame to provide synchronization with the input image.*

The following registers are used to configure this block. These registers are all members of the **Crop** group.

Register	Description
Enable crop	Crop enable: 0 = Off, 1 = On.
start x	Horizontal offset from left side of image in pixels for output crop window.
start y	Vertical offset from top of image in lines for output crop window.
size x	Width of output crop window.
size y	Height of output crop window.

Note: *Please refer to the ISP_CONFIG_PING: fr crop, ISP_CONFIG_PONG: fr crop, ISP_CONFIG_PING: ds crop, and ISP_CONFIG_PONG: ds crop sections in the Mali-IV009 register map for the detailed register descriptions.*

2.7.28 Down scaler

This module is a multi-tap polyphase scaler capable of applying arbitrary scaling factors in horizontal and vertical directions in the shrink direction. The maximum downscale factor is 16.

By default, the scaler is configured with 7 taps and 32 phases in both horizontal and vertical directions. There are 16 coefficient sets available in each direction which can be preconfigured to support a wide range of scaling factors. Switching between coefficient sets allows scaling factors to be altered dynamically without artefact generation.

Scaler Coefficients are calculated using PC tools. A 32 bit LUT with 512 entries is used to store the 8 coefficient sets as shown in the table below. Each coefficient is stored as an 8 bit value.

In the multi-context operation, the 8 coefficients can be shared between all the contexts. A set of coefficients can be selected through the HFILT COEFSET/VFILT COEFSET registers.

Table Index	Set	Phase	31:24	23:16	15:8	7:0
0	0	0	Coefficient3	Coefficient2	Coefficient1	Coefficient 0
1	0	0	0	Coefficient6	Coefficient5	Coefficient4
2	0	1	Coefficient3	Coefficient2	Coefficient1	Coefficient 0

Table Index	Set	Phase	31:24	23:16	15:8	7:0
3	0	1	0	Coefficient6	Coefficient5	Coefficient4
..repeated until...						
62	0	31	Coefficient3	Coefficient2	Coefficient1	Coefficient 0
63	0	31	0	Coefficient6	Coefficient5	Coefficient4
64	1	0	Coefficient3	Coefficient2	Coefficient1	Coefficient 0
65	1	0	0	Coefficient6	Coefficient5	Coefficient4
66	1	1	Coefficient3	Coefficient2	Coefficient1	Coefficient 0
67	1	1	0	Coefficient6	Coefficient5	Coefficient4
..repeated until...						
511	15	31	Coefficient3	Coefficient2	Coefficient1	Coefficient 0
512	15	31	0	Coefficient6	Coefficient5	Coefficient4

2.7.28.1 Key parameters

Register	Description
VFILT_TINC	Vertical scale factor = vertical input height/vertical output height. This is in 4.20 number format.
HEIGHT_IN	Height of frame input in lines.
OHEIGHT	Height of frame output in lines.
VFILT_COEFSET	4 bit parameter used to select one of 16 available vertical scaler coefficient sets.
HFILT_TINC	Horizontal scale factor = horizontal input height/horizontal output height. This is in 4.20 number format
WIDTH_IN	Width of frame input in pixels
OWIDTH	Width of frame output in pixels
HFILT_COEFSET	4 bit parameter used to select one of 16 available horizontal scaler coefficient sets.

Note: *Please refer to the **ISP_CONFIG_PING: ds scaler** and **ISP_CONFIG_PONG: ds scaler** sections in the **Mali-IV009 register map** for the detailed register descriptions.*

2.7.29 Sharpen

This block control the sharpening in RGB domain after demosaicing.

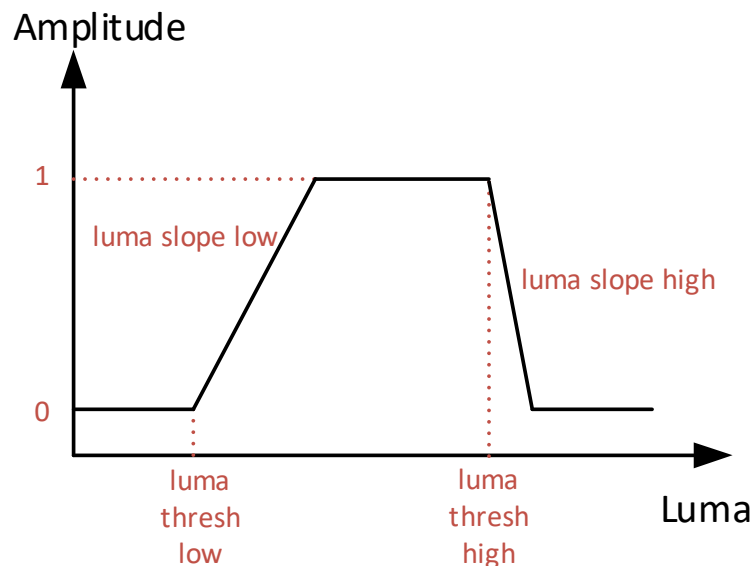
It provides:

- Low boosting of very high frequency. It increases dot noise or unnatural overshoot.
- Low jaggings, Low “worms” artefact.

The amplitude of sharpen is controlled by luminance. This is done by decreasing the amplitude of:

- The dark area using the configuration register, luma thresh low and luma slope low.
- The bright area using the configuration register, luma thresh high and luma slope high.

The following figure shows this effect.



Note: Please refer to the *ISP_CONFIG_PING: fr sharpen*, *ISP_CONFIG_PONG: fr sharpen*, *ISP_CONFIG_PING: ds sharpen*, and *ISP_CONFIG_PONG: ds sharpen* sections in the *Mali-IV009 register map* for the detailed register descriptions.

2.7.30 RGB gamma

This module encodes the output gamma and is typically set to match BT.709 or sRGB gamma curves.

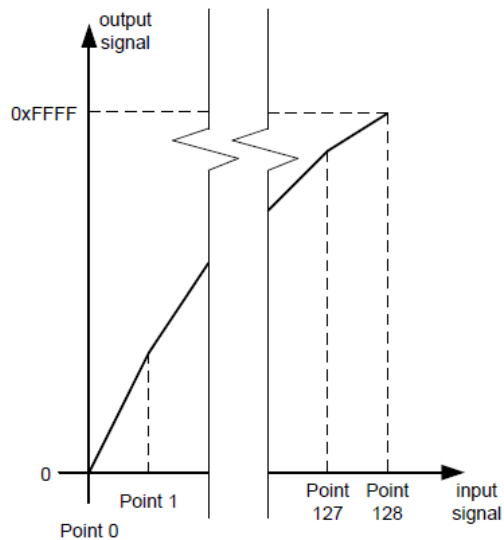


Figure 30. Gamma LUT

This module applies a gamma LUT separately for each of the three (R,G,B) color channels.

In typical configurations, the LUT has 129 evenly spaced nodes labeled 0...128 and the hardware applies linear interpolation between these nodes.

Each data value is a 16bit unsigned number, so it is expected that $\text{Gamma}[0]=0$ and $\text{Gamma}[128]=0xFFFF$, with the other 127 values defining the Gamma correction curve.

Note: *The adaptive contrast enhancement is performed dynamically by the Iridix module. This LUT should be programmed statically based on the desired output gamma characteristics*

Note: *Please refer to the ISP_CONFIG_PING: fr gamma rgb, ISP_CONFIG_PONG: fr gamma rgb, ISP_CONFIG_PING: ds gamma rgb, and ISP_CONFIG_PONG: ds gamma rgb sections in the Mali-IV009 register map for the detailed register descriptions.*

2.7.31 Color space conversion

This module optionally converts the input {R, G, B} pixel values into {Y, U, V} values with a standard 3x3 matrix multiply and a vector offset. If the conversion is not activated, the ISP outputs pixels in RGB format.

$$\begin{pmatrix} Y \\ U \\ V \end{pmatrix} = \begin{pmatrix} Coeffft11 & Coeffft12 & Coeffft13 \\ Coeffft21 & Coeffft22 & Coeffft23 \\ Coeffft31 & Coeffft23 & Coeffft33 \end{pmatrix} \cdot \begin{pmatrix} R \\ G \\ B \end{pmatrix} + \begin{pmatrix} Coeffft01 \\ Coeffft02 \\ Coeffft03 \end{pmatrix}$$

The parameters can be reprogrammed to provide a different transform if required.

Chroma sub-sampling is also available. If selected, this halves the chroma bandwidth from 4:4:4 to 4:2:2. This is a common requirement for H.264 encoders. It is also possible to sub-sample vertically to further reduce the chroma bandwidth to 4:2:0. This last feature is optional and may not be present in all ISP implementations.

The default parameters will provide conversion to BT.709 as shown below.

$$\begin{pmatrix} Y \\ U \\ V \end{pmatrix} = \begin{pmatrix} 0.184 & 0.613 & 0.063 \\ -0.102 & -0.340 & 0.438 \\ 0.438 & -0.398 & -0.039 \end{pmatrix} \cdot \begin{pmatrix} R \\ G \\ B \end{pmatrix} + \begin{pmatrix} 0.063 \\ 0.5 \\ 0.5 \end{pmatrix}$$

2.7.31.1 Key parameters

Register	Description
Enable_Matrix	Set to 1 to enable YUV 4:4:4 conversion
Enable_Filter	Set to 1 to enable lanczos filtering prior to horizontal sub-sampling
Enable_Downsample (Enable_Horizontal_Downsample when the 4:2:0 option is available)	Sub-sample horizontally to YUV4:2:2 format
Enable_Vertical_Downsample	Sub-sample 4:2:2 vertically to YUV 4:2:0 format
Coeffft11, Coeffft12, Coeffft13, Coeffft21, Coeffft22, Coeffft23, Coeffft31, Coeffft32, Coeffft33	Coefficients in the 3x3 conversion matrix. These are 16 bit values in 8.8 number format. The MSbit is a sign bit and is set to 1 if the value is negative.
Coeffto1, Coeffto2, Coeffto3	Offset coefficients
Clip min Y	Minimal value for Y. Values below this value are clipped.
Clip max Y	Maximal value for Y. Values above this value are clipped.
Clip min UV	Minimal value for Cb, Cr. Values below this value are clipped
Clip max UV	Maximal value for Cb, Cr. Values above this value are clipped

Note: *Please refer to the **ISP_CONFIG_PING: fr cs conv**, **ISP_CONFIG_PONG: fr cs conv**, **ISP_CONFIG_PING: ds cs conv**, and **ISP_CONFIG_PONG: ds cs conv** sections in the **Mali-IV009 register map** for the detailed register descriptions.*

2.7.32 3A metering statistics

3A statistics consist of Auto white balance, Auto focus and Auto exposure.

The AF module computes the statistical sharpness value in the image. This value/statistics is used by the software to adjust the lens to have the best focus in region of interest (ROI). This module calculates the edge values, both for the zone and the overall image.

The AWB module collects the stats which are used by the SW for white balancing. It accumulates Zone-wise R/G and B/G stats. It also accumulates the cumulative full frames stats.

Auto exposure stats are used for tuning the sensor exposure. This is done by collecting 5 bin histograms.

The ISP provides programmable tag points for various metering modules. The following figure shows the various tagging points.

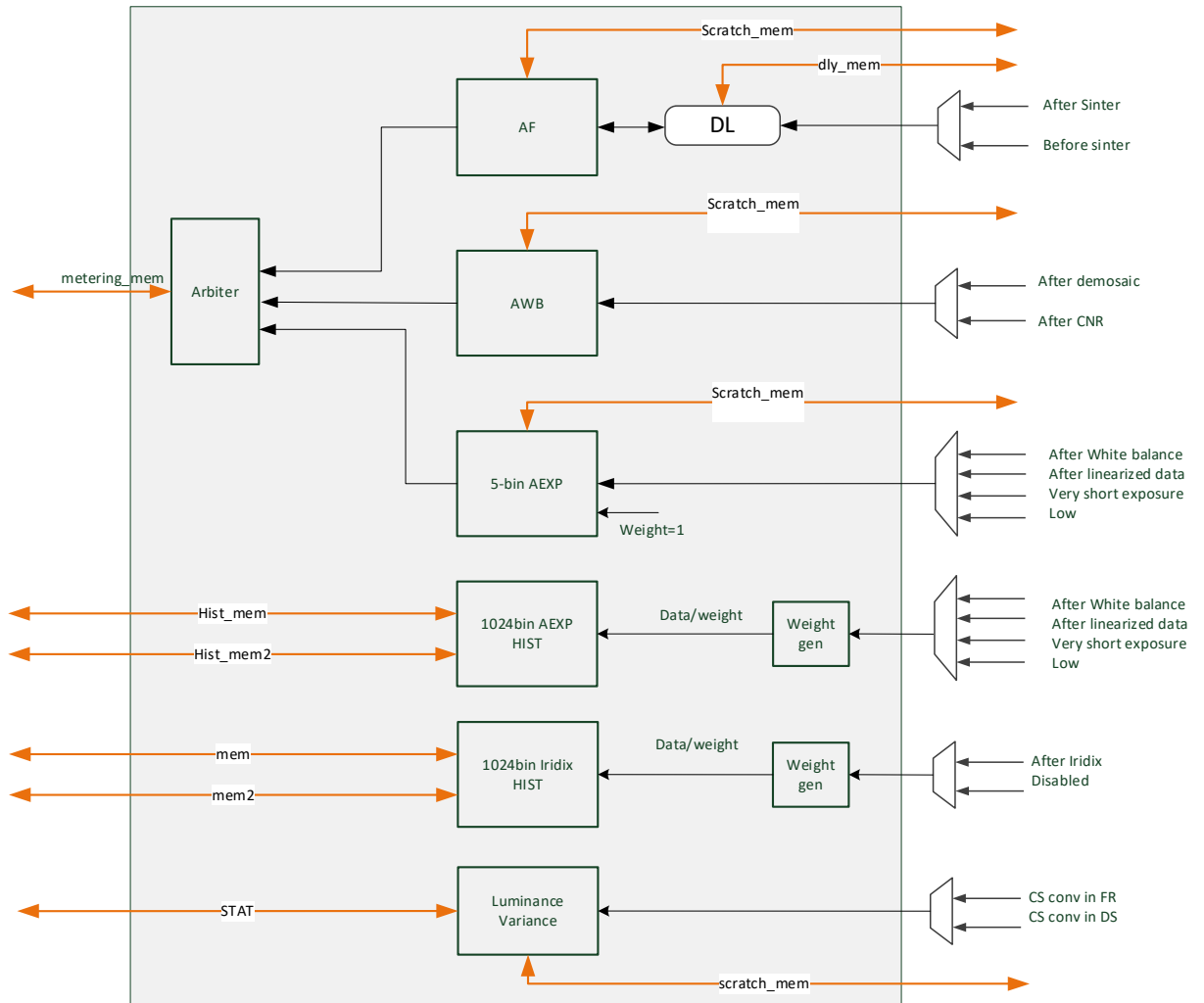


Figure 31. Programmable tagging points

Note: *For the detailed description, please refer to the **ISP_CONFIG_PING:top** and **ISP_CONFIG_PONG:top** registers in the Mali-IV009 register map.*

2.7.32.1 Auto white balance

Color channel correction for white balance is controlled by the AWB firmware algorithm based on AWB statistics. This correction factor is static throughout the frame and updated on frame boundaries as part of the configuration.

AWB statistics can be provided as the zone weighted average of color ratios for the entire image. The **Statistics: AWB stats mode** register can be used to select AWB stats to be

presented as either Red/Green and Blue/Green ratios or Green/Red and Green/Blue ratios for each zone in a 15 x 15 grid. These ratios are in unsigned 4.8 number format, are stored in SRAM, and can be read back through registers **Statistics: AWB RG** and **Statistics: AWB BG**.

AWB statistics may be collected from multiple tap-points throughout the ISP, and AWB includes a configurable decompanding function. Configuration registers allow exclusion of outlier pixel values.

Additional adjustments can be made to exclude pixels in the statistics based on color ratio. In the diagram below, only pixels with valid color ratios bounded by the **Statistics: Cr Ref Min AWB**, **Statistics: Cr Ref Max AWB**, **Statistics: Cb Ref Min AWB**, and **Statistics: Cb Ref Max AWB** registers are used in the statistics. These registers set the minimum and maximum Red/Green and Blue/Green ratios. The diagram shows the Planckian locus and the region bounded by these registers in color ratio space (note that the plot is B/G vs R/G).

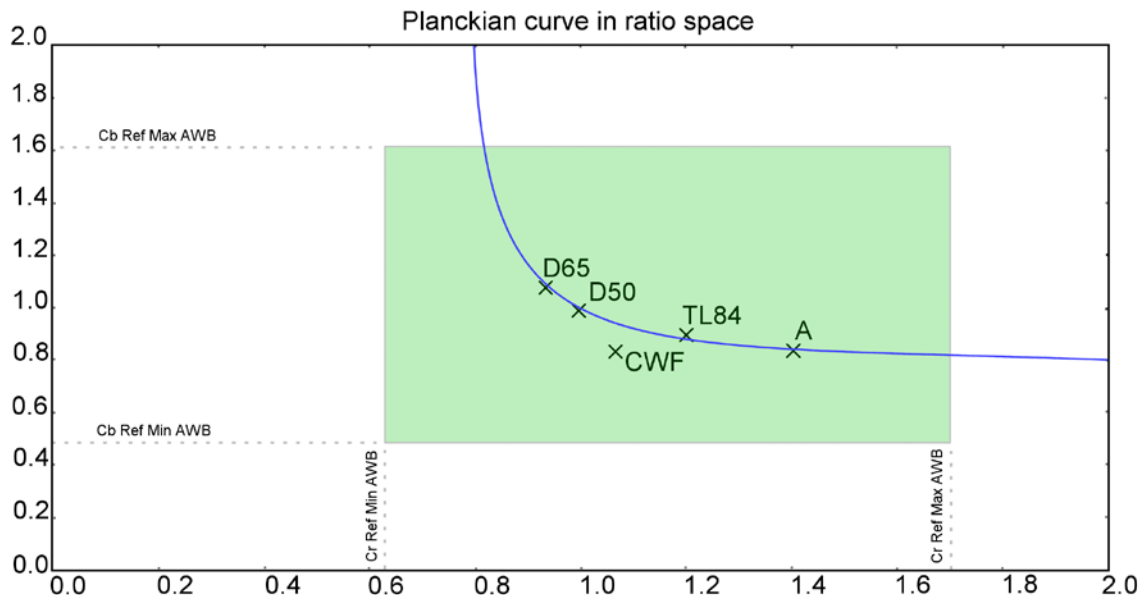


Figure 32. Boundaries for statistics in color ratio space

The boundaries of the values included in the statistics can be further trimmed by the use of **Statistics: Cb Ref Low AWB**, **Statistics: Cb Ref High AWB**, **Statistics: Cr Ref Low AWB** and **Statistics: Cr Ref High AWB**. The values in these registers set the locations of the two triangular areas on the graph shown below.

Note that in this example the values of the **Low** registers are lower than the values of the **High** registers. This may not always be the case, depending on the required position of the 'low' and 'high' triangular areas excluded from the statistics.

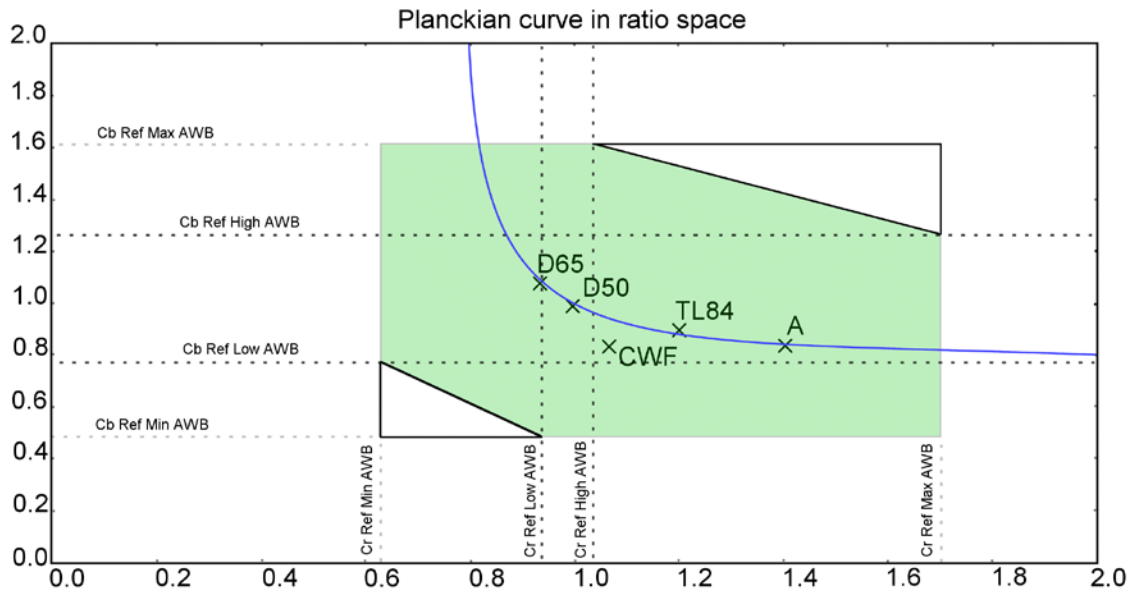


Figure 33. Additional boundaries for statistics in color ratio space

Additional registers, **Statistics: White Level AWB** and **Statistics: Black Level AWB**, set the upper and lower intensity boundaries within which pixels contribute to the statistics. The total number of pixels used in for the AWB statistics can be read from the **Statistics: AWB SUM** register.

The following registers are used to configure this block. These registers are all members of the **Statistics** group:

Register	Description
AWB stats mode	0: Statistics collected as Green/Red and Blue/Red ratios. 1: Statistics collected as Red/Green and Blue/Green ratios.
AWB RG	Green/Red or Red/Green color ratio from zone weighted average in unsigned 4.8 fixed point format. Read only.
AWB BG	Green/Blue or Blue/Green color ratio from zone weighted average in unsigned 4.8 fixed point format. Read only.
SUM	Read only register indicating the total number of pixels used in the AWB statistics.
Cr Ref Min AWB	Specifies the minimum Red/Green color ratio of a white region in 4.8 format that will be included in AWB statistics.
Cr Ref Max AWB	Specified the maximum Red/Green color ratio of a white region in 4.8 format that will be included in AWB statistics.
Cb Ref Min AWB	Specifies the minimum Blue/Green color ratio of a white region in 4.8 format that will be included in AWB statistics.

Register	Description
Cb Ref Max AWB	Specifies the maximum Blue/Green color ratio of a white region in 4.8 format that will be included in AWB statistics.
White Level AWB	Specifies the maximum pixel intensity value that will be included in the AWB statistics.
Black Level AWB	Specifies the minimum pixel intensity value that will be included in the AWB statistics.

Note: *Please refer to the **ISP_CONFIG_PING: metering awb** and **ISP_CONFIG_PONG: metering awb** sections in the Mali-IV009 register map for the detailed register descriptions.*

2.7.32.2 AWB zone data

AWB statistics for each zone can also be indirectly accessed in firmware. As in the global case, the average Green/Blue or Blue/Green, average Green/Red or Red/Green values, and the number of pixels used in the AWB statistics can be read back on a per-zone basis. Color ratios are in unsigned 4.8 format while the total number of pixels is in integer format. The following table illustrates the data structure for m x n zones.

Table Index	Zone	31:28	27:24	23:20	19:16	15:12	11:8	7:4	3:0
0	1		Average Green/Blue [27:16]				Average Green/Red [11:0]		
1	1	Number of pixels used for AWB							
2	2		Average Green/Blue [27:16]				Average Green/Red [11:0]		
3	2	Number of pixels used for AWB							
repeated until ...									
2mn-2	mn		Average Green/Blue [27:16]				Average Green/Red [11:0]		
2mn-1	mn	Number of pixels used for AWB							

Figure 34. Data structure for m x n zones

2.7.32.3 Auto Focus

Auto Focus statistics are composed of Region-Of-Interest (ROI) or zonal and normalized Full image multi-direction contrast metrics. This contrast metrics is used by the CPU outside of the ISP core to determine the position of the lens to achieve the best possible focus.

For the AF block, zones are SW configurable. The Zonal Contrast metrics is calculated for each pixel and accumulated over the full zone. For each pixel, contrast is calculated along 4 directions.

Apart from this, angular direction of diagonal lines can also be controlled using a kernel select config parameter as in table1. To improve the response in low light and low pass imaging scenarios, the calculated contrast is of the forth order (quartic contrast sum).

These zonal metrics are not weighted in the hardware, however the software can apply the zone-based weights post computation.

Figure 35 shows that optimal focus is achieved when the AF contrast metric is at its highest point.

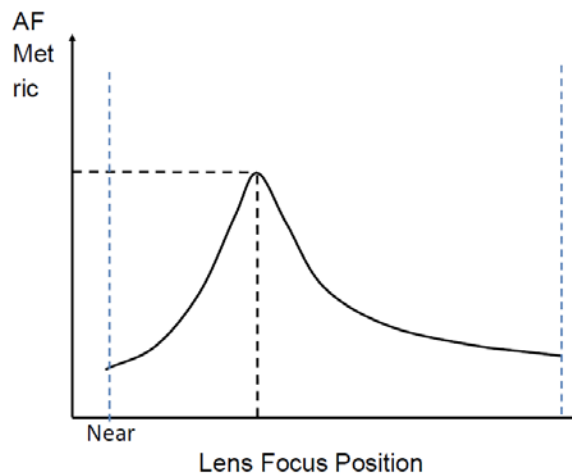


Figure 35. AF Metric response with respect to lens position

Note: *Every time the CPU moves the lens, the AF metrics are re-computed and analyzed for a given ROI.*

2.7.32.3.1 AF metrics data computation

The zonal accumulated contrast metrics are stored in floating point format with 16b mantissa and 5b exponent. Apart from contrast metrics we accumulate squared image and quartic image data over the zone.

Zonal data packing

Each zone accumulates the statistics shown in the following table:

Statistics Name	Data Format	Description
I2	16b Mantissa, 5b Exponent	Zonal squared image pixel sum
I4	16b Mantissa, 6b Exponent	Zonal quartic image pixel sum
E4	16b Mantissa, 5b Exponent	Zonal multi-directional quartic edge sum

Table 4. Zone statistics

Bits	31:16			15:0	
Register1	I4 Mantissa [15:0]			I2 Mantissa [15:0]	
Bits	31	30:26	25:21	20:16	15:0
Register2	I4 exp [5]	E4 exp [4:0]	I4_exp [4:0]	I2 exp [4:0]	E4 Mantissa [15:0]

Table 5. Zonal data packing 2 x 32b register

Zone 1	Register1 [31:0]
	Register2 [31:0]
Zone 2	Register1 [31:0]
	Register2 [31:0]
....

Zone N – 2	Register1 [31:0]
	Register2 [31:0]
Zone N – 1	Register1 [31:0]
	Register2 [31:0]

Table 6. AF Zone data stored in internal memory

Global Contrast Statistics: Apart from zonal statistics, AF also accumulates normalized quartic edge sum. Its stored in a 32b register.

2.7.32.4 Auto Exposure

Auto Exposure (AE) statistics are collected immediately after the Gain and offset block after white balance and ISP gain is applied. Two histogram types are available for metering:

- 5-bin local and global histogram
- 1024-bin global histogram

2.7.32.4.1 5-bin histogram

A 5-bin normalized histogram is generated for each zone and for the entire image with adjustable histogram bin boundaries. The parameters **Statistics_Hist_Thresh [i] [j]** are set to define the threshold intensity values between each bin i and j.

The registers **Statistics_Hist [i]** provide the global normalized pixel counts for each bin i. The total sum is normalized to 0xFFFF.

The centre bin of the histogram is not available and can be calculated in firmware by subtracting the values of Hist0, Hist1, Hist3, and Hist4 from 0xFFFF as in the following equation:

$$Hist_2 = 0xFFFF - \sum_i Hist_i$$

where $i = 0, 1, 3, \text{ and } 4$.

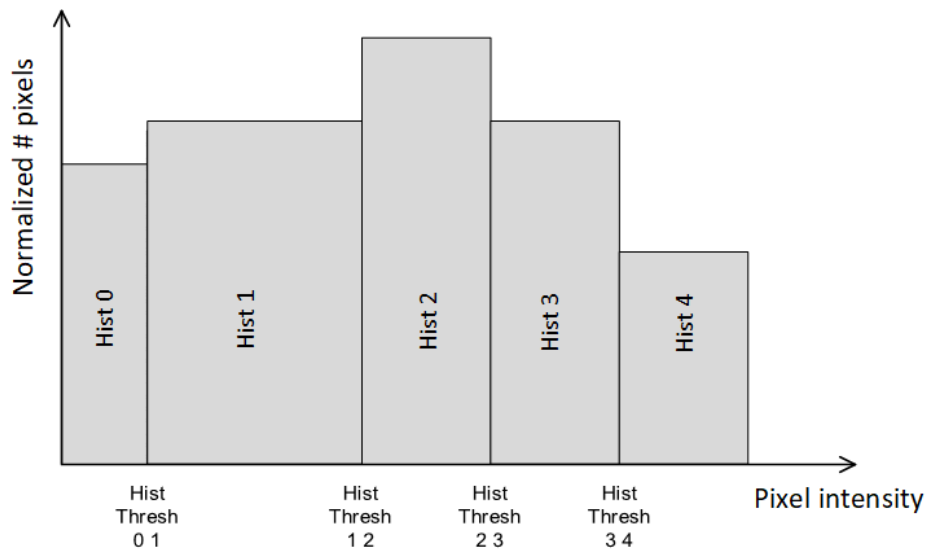


Figure 36. 5-bin Auto-Exposure Histogram

The zone by zone data collected by the AE statistics module is not directly available in the register space. Indirect access to all the “metering data” statistics is available in the firmware.

Note: *The zone statistics values are stored in internal memory, and are updated by the ISP hardware. This memory is read only.*

An internal table with Histx data gives the normalised values of the histograms for each of the zones as shown in the table below for mxn zones. The order of the zones is in raster order starting from the top left corner of the image. The sum of the histogram data is normalized to 0xFFFF for each zone. Note that as in the global histogram, the Hist2 is not available but can be calculated based on the normalized sum and the values of Hist0, Hist1, Hist3 and Hist4 for each zone.

Table Index	Zone	31:28	27:24	23:20	19:16	15:12	11:8	7:4	3:0
0	1	Hist 1				Hist 0			
1	1	Hist 4				Hist 3			
2	2	Hist 1				Hist 0			
3	2	Hist 4				Hist 3			
repeated until ...									
2mn-2	mn	Hist 1				Hist 0			
2mn-1	mn	Hist 4				Hist 3			

2.7.32.4.2 1024-bin histogram

A separate global 1024-bin histogram is built for the entire image as shown in [Figure 37](#). The global histogram data is not directly available in register space but can be indirectly accessed in the firmware. The values are double buffered and remain stable for the duration of the entire frame after the one from which they were derived. The global histogram is also weighted by zone but is not normalized. It is expected that normalization of the statistics data be performed in the firmware. This is available to the firmware as a consecutive array of 1024x32 bit unsigned integers.

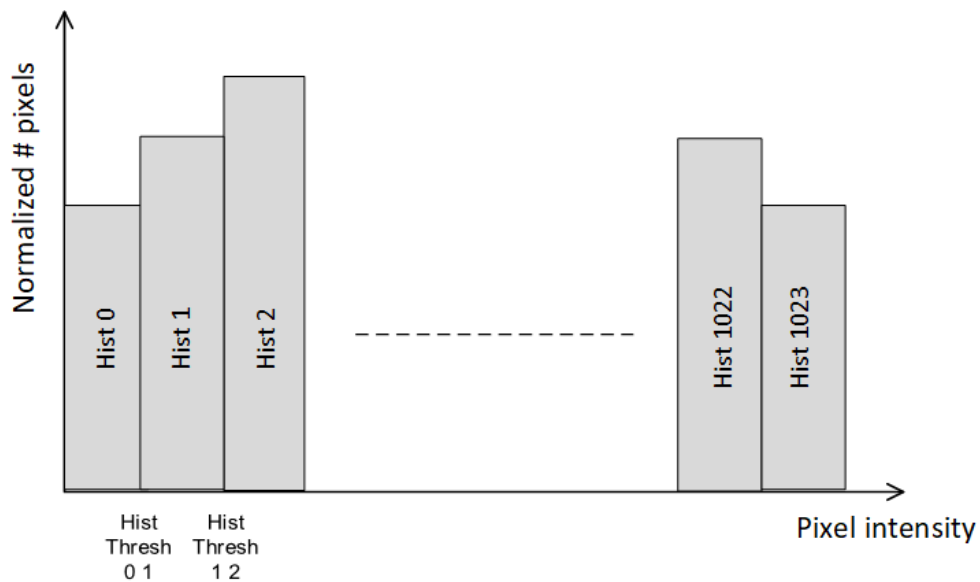


Figure 37. 1024-bin Auto-Exposure histogram

Table Index	31:28	27:24	23:20	19:16	15:12	11:8	7:4	3:0
0	Hist 0							
1	Hist 1							
2	Hist 2							
repeated until...								
1022	Hist 1022							
1023	Hist 1023							

2.7.32.5 Mean luminance and variance of mean luminance

The ISP provides a stat of zonal luminance mean and the variance of the zonal luminance mean. The module always uses a fixed zone size of 32x16 (horizontal x vertical) and the minimum frame resolution required for this stat module to give usable result is 512x256.

The stats are stored in SRAM of 512 locations. Each location contains 10-bit (LSBs) of mean information and 12 bit (MSBs) of the variance of the luminance mean information for each zone.

3 Interrupts

This section describes the Mali-IV009 ISP interrupt behaviour.

It provides information about issues:

- concerning the hardware interface to the MCU interrupt mechanism that a system designer must be aware of when integrating the Mali-IV009 ISP.
- that programmers must be aware of when writing interrupt service routines.

The ISP provides a level or pulse interrupt, which is based on the pulse mode setting. The MCU can select a set of registers by programming the appropriate mask register.

The following table lists the various ISP interrupts:

Bit	Name	Description
[0]	Isp_start	When the first pixel comes out of input_port, the ISP asserts this interrupt.
[1]	Isp_done	When the ISP finishes all its internal processing, processing including the iridix filtering and the metering stats calculation, it asserts this interrupt. This doesn't ensure that the DMA writer has finished the storing of current frame (there are separate interrupt sources for this).
[2]	Mult-context management error	In the multi-context mode, when a new frame comes before the config_done for that address space, the ISP interrupts with this bit. This interrupt is not available in the single context mode.
[3]	Broken frame error	This is a frame monitor error interrupt. This indicates that the ISP has detected a frame dimension error. The frame monitor checks the incoming frames height, width, horizontal blanking and vertical blanking against the programmed values. If there is any mismatch, ISP provides this interrupt. When this bit is set, the MCU must read the frame monitor status register.
[4]	Metering AF done	Auto-focus metering has completed storing all its stats. Once this interrupt is set, the AF stats address space is ready to be read.
[5]	Metering aexp done	Auto-exposure metering has completed storing all its stats. Once this interrupt is set, the AEXP stats

Bit	Name	Description
		address space is ready to be read.
[6]	Metering awb done	Auto-white balance metering has completed storing all its stats. Once this interrupt is set, the AWB stats address space is ready to be read.
[7]	Metering aexp 1024 bin hist done	When the 1024-bit aexp histogram stats are updated, ISP interrupts with this bit.
[8]	Iridix Metering histogram done	When the 1024-bit post Iridix histogram stats are updated, ISP interrupts with this bit.
[9]	Lut init done	This interrupt occurs when the ISP has completed all LUT memory initialization. NOTE: This is not needed in production systems. This is mostly used for simulation environments.
[10]	FR y-dma write done	When the full resolution DMA completes storing the current frame, it gives this interrupt.
[11]	FR uv-dma write done	When the full resolution UV DMA completes storing the current frame, it gives this interrupt.
[12]	DS y-dma write done	When the downscale pipeline DMA completes storing the current frame, it gives this interrupt.
[13]	DS uv-dma write done	When the downscale pipeline UV DMA completes storing the current frame, it gives this interrupt.
[14]	Linearization done	When the last module in linearize cluster completed processing the current frame, ISP issues this interrupt.
[15]	Static dpc done	When the last module in frontend cluster completed processing the current frame, ISP issues this interrupt.
[16]	Ca correction done	When the last module in noise reduction cluster completed processing the current frame, ISP issues this interrupt.
[17]	Iridix done	When the last module in tone mapping cluster completed processing the current frame, ISP issues this interrupt.
[18]	3D lut done	When the last module in bayer2 rgb cluster completed processing the current frame, ISP issues this interrupt.
[19]	Watchdog timer	This interrupt is triggered when the watchdog timer times out. The watchdog timer counts the interval between the frame start and frame end. The expected max interval is programmable.
[20]	Frame collision interrupt	This interrupt is set when a new frame comes in but the ISP is still busy with the previous frame.
[21]	Luminance variance done	When the luminance variance metering module

Bit	Name	Description
		completes the stats update, the ISP provides this interrupt.
[22]	DMA error interrupt	When any of the internals ISP DMAs detect any interrupt, ISP sets this bit. Please refer to the <code>ISP_COMMON:isp_global_monitor</code> for details.
[23]	Input port safely stopped	When the input port mode request changes to <code>SAFE_STOP</code> and input port actually stops, this interrupt is generated.

3.1 Mask vector register

The MCU can mask a particular interrupt by setting its interrupt mask bit to 1. By default all the interrupts are enabled. The change of mask vector affects the coming in events and but also affect the events which are in the shadow registers. For example, if the particular mask bit is set to 1, the related shadow register bit is cleared. Due to the synchronous design, the change of mask will affect the interrupt output by the next cycle.

3.2 Status vector register

The status register shows the current status of the ISP interrupts. This is a sticky register and must be cleared by the MCU.

3.3 Clear vector register

The MCU clears the interrupt bitwise. When a bit is set to 1, then the corresponding bit in the status register is cleared only if another interrupt from the same source is pending. In that case, status register will still be 1.

3.4 Interrupt clear

This is a qualifier to the clear vector register. Once the MCU writes to the clear vector register, it needs to generate a pulse in this clear register.

3.5 Shadow structure

The interrupt handler supports the shadowing structure with the depth of 1 for each coming interrupt event. For each vector bit of interrupt events, it can be shadowed or not based on the shadow disable vector setting. When being shadowed, the particular occurring interrupt event is stored in the shadow bit. If the related status bit is not cleared the it means the previous event for this vector bit is still pending.

3.6 Level or pulse mode

The interrupt handler supports level or pulse mode based on the pulse mode setting. It outputs the new level request or interrupt pulse, only when the previous one has been served by the clearing action from the MCU.

4 Clocking, resets and power management

This section describes the clocking, reset and power management options available with the Mali-IV009 ISP.

4.1 Clock and reset

The Mali-IV009 has the following clock and reset domains:

- vclk-rstn
- aclk-aresetn

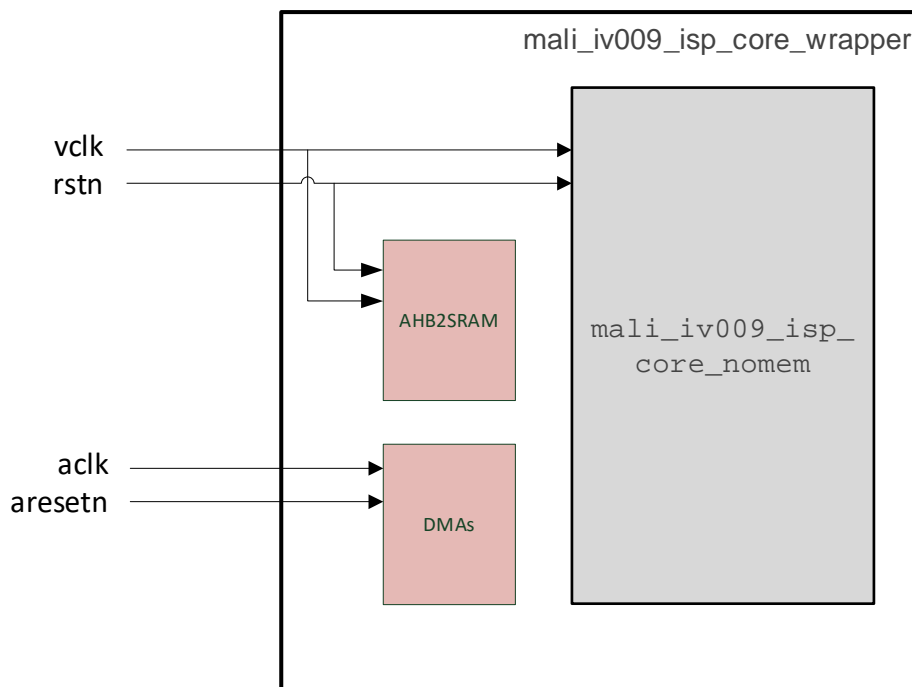


Figure 38. ISP clock and reset domain

All the resets are asynchronous and their assertion is asynchronous. However, the de-assertion **must be** synchronous to the vclk and aclk respectively. Refer to [Figure 39](#).

The ISP does not require any particular reset sequence for the occurrence of aresetn and rstn. They can occur in any order. But the following conditions must be satisfied:

- All the reset sequencing must be completed before any ISP configuration starts.
- Both the resets must be active together.

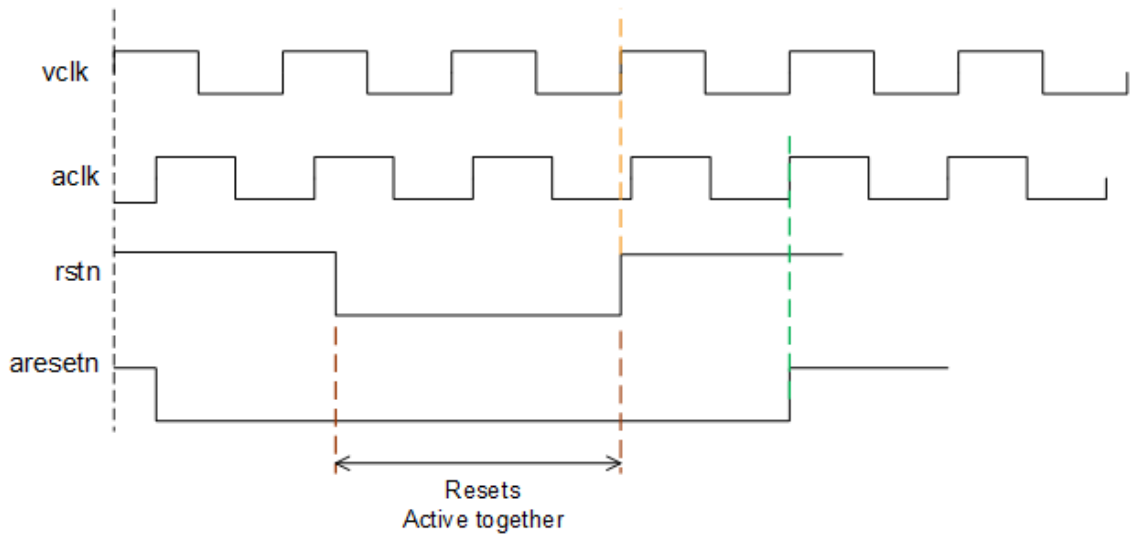


Figure 39. Clock and reset timing diagram

NOTE: *It is recommended that the SoC must have an external reset control mechanism such that the system level soft reset can be driven to the ISP primary rstn and aresetn pins also. This can be very useful if the system detects error or hung condition and the ISP needs to be reset separately.*

4.2 Stalling the video pipeline processing

The clock enable signal (`vcke`) can be used to stall the video pipeline processing. Any AXI domain processing is unaffected by the `vcke`. During the time that the clock enable signal is de-asserted the IP does not accept input video data or move data down the video pipeline. It maintains the current state of the video pipeline.

When you use `vcke` to stall the IP processing you must consider the following requirements related to stalled horizontal blanking and flushing of the pipeline:

- **Stalled horizontal blanking:** At least 32 cycles (pixel clock cycles) of the horizontal blanking which is not stalled is required. See [Figure 40](#).
- **Flushing of the pipeline:** When the end of a frame occurs at the start of the pipeline there are still many lines of unprocessed video data within the pipeline. The `vcke` must be asserted to move this data down the pipeline and 'Flush' it, so cycles where `vcke` is de-asserted will stall the flushing of the pipeline. The pipeline will have completed flushing when the ISP end interrupt is asserted.

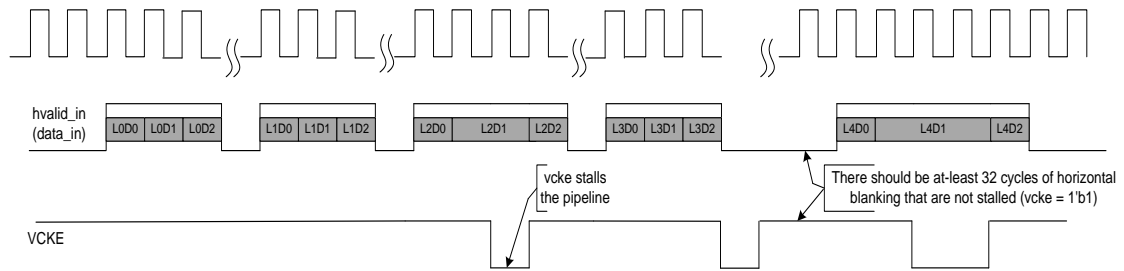


Figure 40. Interface Requirements with Stalled Pipeline Functionality

4.3 Low power features

4.3.1 Manual module level clock gating

The Mali-IV009 ISP has 2 levels of clock gating.

- The level-2 clock gating is inferred clock gating, which the synthesis tool will insert.
- The level-1 is a manual clock gating. Manual clock gating is applied to selective modules. Please refer to the **ISP_COMMON:isp global lp** registers for details.

The Mali-IV009 has the following registers to control its clock gating behaviour:

- CLK disable register
- CG disable register

4.3.1.1 CLK disable register

This register is provided to statically gate the clock for all the DMAs. This must be static signal and must be used when the corresponding DMA functionality is not needed.

4.3.1.2 CG disable register

This register is used to disable the clock gate cell for the dynamic clock gate condition. When a bit is set to 1, the corresponding clock will NOT be gated even when it enters into the V-blank region.

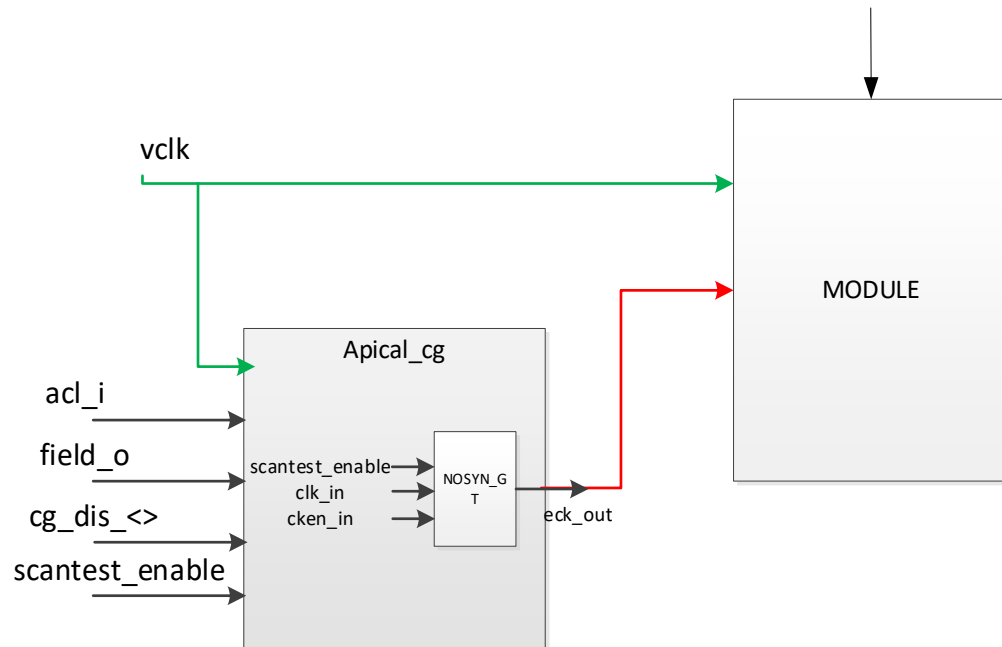


Figure 41. Manual clock gating structure

The synthesisable RTL delivers a clock gate cell module called NOSYN_GT. This cell can be replaced by the technology library clock gate cell, but the interface signal names must be the same.

The clock will be gated for a module under the following conditions:

- When a module enters into the v-blank region.
This is referred to as a dynamic clock gating condition.
- When a module is bypassed – in this case the clock can be gated through register programming.
This is referred to as a static clock gating condition.

Although these two conditions are common, there are some exceptions. The following table describes the clock gating conditions:

Note: *When the ISP is in the scan mode through the interface port **scantest_enable** all the clock gate cells are disabled. However, each clock gate cell can be selectively disabled through its own configuration register as described in the following table.*

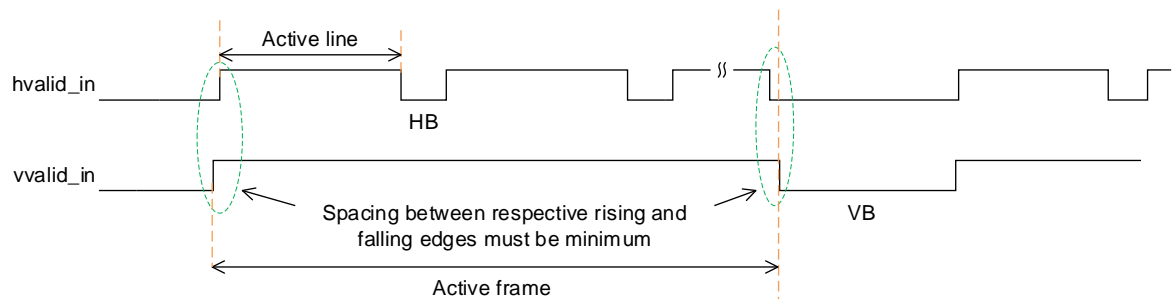
Module name	The Clock is disabled if...	The Clock gate is disabled if...	Comment
Frame stitch	<ul style="list-style-type: none"> the module is in Vertical Blanking (VB) the module is bypassed the Multi-exposure data is not stitched 	cg_dis_<module> is set to 1	
Raw frontend	<ul style="list-style-type: none"> the module is in Vertical Blanking (VB) the module is bypassed 	cg_dis_<module> is set to 1	
Defect pixel	the module is bypassed	cg_dis_<module> is set to 1	
Sinter	<ul style="list-style-type: none"> the module is in Vertical Blanking (VB) the module is bypassed 	cg_dis_<module> is set to 1	
Temper	<ul style="list-style-type: none"> the module is in Vertical Blanking (VB) the module is bypassed 	cg_dis_<module> is set to 1	
Temper DMA (2 reader + 2 writers)	the DMA clock is explicitly disabled through the clk_dis_<DMA name> config signal	the DMA clock is NOT explicitly disabled through clk_dis_<DMA name> config signal	
CA correction	<ul style="list-style-type: none"> the module is in Vertical Blanking (VB) the module is bypassed 	cg_dis_<module> is set to 1	
Radial shading	<ul style="list-style-type: none"> the module is in Vertical Blanking (VB) the module is bypassed 	cg_dis_<module> is set to 1	
Mesh Shading	<ul style="list-style-type: none"> the module is in Vertical Blanking (VB) the module is bypassed 	cg_dis_<module> is set to 1	
Iridix	<ul style="list-style-type: none"> the module is in Vertical Blanking (VB) the module is bypassed 	cg_dis_<module> is set to 1	NOTE: Only clock enable is gated and not the clock. Meaning L1 clock will be gated
Demosaic RGB	<ul style="list-style-type: none"> the module is in Vertical Blanking (VB) 	cg_dis_<module> is set to 1	

Module name	The Clock is disabled if...	The Clock gate is disabled if...	Comment
	<ul style="list-style-type: none"> the module is bypassed cfa_pattern is NOT RGGB 		
Demosaic RGBIr	<ul style="list-style-type: none"> the module is in Vertical Blanking (VB) the module is bypassed cfa_pattern is NOT RGBIr 	cg_dis_<module> is set to 1	
PF correction	<ul style="list-style-type: none"> the module is in Vertical Blanking (VB) the module is bypassed 	cg_dis_<module> is set to 1	
CNR	<ul style="list-style-type: none"> the module is in Vertical Blanking (VB) the module is bypassed 	cg_dis_<module> is set to 1	
3D LUT + Non-eq gamma	<ul style="list-style-type: none"> the module is in Vertical Blanking (VB) the module is bypassed 	cg_dis_<module> is set to 1	
RGB scaler	<ul style="list-style-type: none"> the module is bypassed the downscale pipeline is disabled 	cg_dis_<module> is set to 1	Scaler doesn't have any dynamic clock gating
RGB Gamma FR	<ul style="list-style-type: none"> the module is in Vertical Blanking (VB) the module is bypassed the downscale pipeline is disabled 	cg_dis_<module> is set to 1	
RGB Gamma DS	<ul style="list-style-type: none"> the module is in Vertical Blanking (VB) the module is bypassed the downscale pipeline is disabled 	cg_dis_<module> is set to 1	
RGB Sharpen FR	<ul style="list-style-type: none"> the module is in Vertical Blanking (VB) the module is bypassed the downscale pipeline is disabled 	cg_dis_<module> is set to 1	

Module name	The Clock is disabled if...	The Clock gate is disabled if...	Comment
RGB Sharpen DS	<ul style="list-style-type: none"> the module is in Vertical Blanking (VB) the module is bypassed the downscale pipeline is disabled 	cg_dis_<module> is set to 1	

Table 7. Clock and clock gate disable conditions

The following figure shows the ways in which power saving can be maximised by properly timing the `hvalid_in` and `vvalid_in` signals:



HB: Horizontal blanking period, should be as small as possible

VB: Vertical blanking period, should be as large as possible

Figure 42. Optimal HB and VB

Referring to [Figure 42](#):

- The Horizontal blanking period should be as small as possible.
- The Vertical blanking period should be as large as possible.
- The difference between the rising and falling edges of the `hvalid_in` and `vvalid_in` signals should be as small as possible.
- Vertical blanking should start immediately after the last pixel is sent to the ISP.

5 Error and deadlock considerations

The ISP provides multiple interrupts when it detects various error conditions. Once the MCU detects an error interrupt, the following steps must be followed to re-initialize the core:

1. Read the status register to know the exact source of the error interrupt.
 2. Read the details of the error status register.
 3. Mask all the error interrupts.
 4. Configure the input port register **ISP_COMMON:input port: mode request** to **safe_stop** mode.
 5. Wait until input port stops. There are 2 ways to know when input port has stopped:
 - a. Read back the **ISP_COMMON:input port: mode status** register to see the status of mode request. And wait until it shows the status as safe stop (value:0).
 - b. Wait for the “input port safely stopped” interrupt.
 6. Assert the global fsm reset.
 7. Clear the global fsm reset.
 8. Skip the 4 **isp_done** interrupts for every context to make sure that the ISP has flushed out all the broken frames and has recovered.
 9. Issue clear alarms to make sure that the previous alarms are cleared.
 - a. Write 1 to all the bits in register 0x00058 {temper dma clr alarm, output dma clr alarm, context error clr, broken frame error clear}.
- | Addr | Mode | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | Default |
|---------|------|----|----|----|----|----|----|----|----|----|----|----|----|-------------------------|-------------------------|----------------------|-----------------------------|------------|
| 0x00058 | RW | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | 0x00000000 |
| | | | | | | | | | | | | | | temper dma
clr alarm | output dma
clr alarm | context
error clr | broken frame
error clear | |
- b. Wait for the DMA alarms to be cleared. Read the register 0x00054. Once this register is all 0, write 0 to the 0x0058 register {temper dma clr alarm, output dma clr alarm, context error clr, broken frame error clear}.
 10. Reconfigure the ISP configuration space.
 11. Unmask the necessary interrupt sources.

12. Read the **ISP_COMMON:ping_pong_config_select** signal to sync up with the software state.
13. Configure the correct PING or PONG configure register space.
14. Configure the input port in safe_start mode.

The following table describes the different error conditions.

Error condition	Interrupt
Broken frame: When the incoming frame geometry is not the same as the programmed geometry, the ISP gives this interrupt.	Bit[3]
Mult-context management error: This is applicable only for the multi-context mode. When the new frame comes before the config_done for that address space, the ISP interrupts with this bit.	Bit[2]
DMA FIFO error: The ISP provides error interrupt when any of the internal DMAs overflow or underflow.	Bit[21]
Watchdog timer: This interrupt is triggered when the watchdog timer is timed out. This timer counts the interval between frame start and frame end. The expected max interval is programmable.	Bit[19]
Frame collision: This interrupt is set when a new frame comes in but the ISP is still busy with the previous frame.	Bit[20]

Note: *Please refer to the **ISP_COMMON:isp global interrupt** and the **ISP_COMMON:isp global monitor registers** for the detailed register descriptions.*

6 DMA output format

Base mode	Plane select	Colour Format	Packing																														
			Bit Number																														
			31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1
0	0	DMA write disabled																															
1	0	RGB32 (also used for RGB666)	0	0	0	0	0	0	0	0	Red (7:0)						Green(7:0)						Blue(7:0)										
2	0	A2R10G10B10, 10 bit (30 bit colour)	0	0	Red (9:0)						Green(9:0)						Blue(9:0)																
3	0	RGB 565	Red (4:0)					Green(5:0)					Blue(4:0)					Red (4:0)					Green(5:0)					Blue(4:0)					
4	0	RGB24	Blue1(7:0)						Red0(7:0)						Green0(7:0)						Blue0(7:0)												
			Green2(7:0)						Blue2(7:0)						Red1(7:0)						Green1(7:0)												
			Red3(7:0)						Green3(7:0)						Blue3(7:0)						Red2(7:0)												
5	0	GEN32	Generic Data(31:0)																														
6	0	RAW16	RAW1(15:0) (smaller sizes msb aligned, zero padded)															RAW0(15:0) (smaller sizes msb aligned, zero padded)															
8	0	AYUV, 8 bit 4:4:4	0	0	0	0	0	0	0	0	Y(7:0)						U(7:0)						V(7:0)										
9	0	Y410, 10bit 4:4:4	0	0	V(9:0)						Y(9:0)						U(9:0)																
10	0	YUY2, 8 bit 4:2:2	V0(7:0)						Y1(7:0)						U0(7:0)						Y0(7:0)												
11	0	UYVY, 8 bit 4:2:2	Y1(7:0)						V0(7:0)						Y0(7:0)						U0(7:0)												
12	0	Y210, 10bit 4:2:2	U0(9:0)									0	0	0	0	0	0	Y0(9:0)									0	0	0	0	0	0	
			V0(9:0)									0	0	0	0	0	0	Y1(9:0)									0	0	0	0	0	0	
13	0	NV12/21 Y	Y3(7:0)						Y2(7:0)						Y1(7:0)						Y0(7:0)												
	1	NV12 U/V	V1(7:0)						U1(7:0)						V0(7:0)						U0(7:0)												
	2	NV21 U/V	U1(7:0)						V0(7:0)						U0(7:0)						V0(7:0)												
17	0	Y plane 10 bit 4:2:2/4:2:0	Y3(1:0)	Y2(9:0)								Y1(9:0)								Y0(9:0)													
			Y6(3:0)				Y5(9:0)								Y4(9:0)								Y3(9:2)										
			Y9(5:0)				Y8(9:0)								Y7(9:0)								Y6(9:4)										
	1	U/V plane 10 bit 4:2:2	V1(1:0)	U1(9:0)								V0(9:0)								U0(9:0)													
			U3(3:0)				V2(9:0)								U2(9:0)								V1(9:2)										
			V4(5:0)				U4(9:0)								V3(9:0)								U3(9:4)										
	2	U/V plane 10 bit 4:2:0	V1(1:0)	U1(9:0)								V0(9:0)								U0(9:0)													
			U3(3:0)				V2(9:0)								U2(9:0)								V1(9:2)										
			V4(5:0)				U4(9:0)								V3(9:0)								U3(9:4)										

Base mode	Plane select	Colour Format	Packing																																										
			Bit Number																																										
			31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0											
19	0	P210/P010 10 bit Y plane 4:2:2/4:2:0	Y1(9:0)										0	0	0	0	0	0	Y0(9:0)										0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
	1	P210 10 bit 4:2:2 U/V plane	V0(9:0)										0	0	0	0	0	0	0	U0(9:0)										0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
			V1(9:0)										0	0	0	0	0	0	0	U1(9:0)										0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	2	P010 10 bit 4:2:0 U/V plane	V0(9:0)										0	0	0	0	0	0	0	U0(9:0)										0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
			V1(9:0)										0	0	0	0	0	0	0	U1(9:0)										0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
20	0	YUV 4:2:2 10bit packed	Y1(1:0)	V0(9:0)										Y0(9:0)						U0(9:0)																									
			V1(3:0)			Y2(9:0)										U1(9:0)						Y1(9:2)																							
			Y4(5:0)					U2(9:0)										Y3(9:0)						V1(9:4)																					
Notes																																													

7 Known limitations

The Arm Mali-IV009 Image Signal Processor (ISP) does not support IP-XACT.