AN13857

i.MX 8M Series MIPI Capture System

Rev. 0 — 15 March 2023

Application note
COMPANY PUBLIC

Document information

Information	Content
Keywords	i.MX 8M, MIPI Capture System, Camera Porting
Abstract	This document shows the differences in the MIPI capture system between the i.MX 8M series processors and provides guidance on how to port new cameras on them.

1 Introduction

This application note provides detailed information about the MIPI capture system on i.MX 8M series processors. It covers the implementation differences between various i.MX processors and provides information on how to port new cameras and debug operations.

This document applies to these i.MX processors:

- i.MX 8M (i.MX 8MQ)
- i.MX 8M Mini (i.MX 8MM)
- i.MX 8M Nano (i.MX 8MN)
- i.MX 8M Plus (i.MX 8MP)

These devices are referred to by their abbreviated names throughout this document (shown above in the parentheses).

1.1 References

- i.MX 8M series reference manuals
- i.MX 8M series datasheets

1.2 Acronyms and abbreviations

Table 1 defines the acronyms and abbreviations used in this document.

Table 1. Acronyms and definitions

Term	Definition
MIPI	Mobile Industry Processor Interface
CSI-2	MIPI Alliance Standard for camera serial interface 2
D-PHY	one of the physical-layer interfaces developed by MIPI
CSI	CMOS Sensor Interface
ISI	Image Sensing Interface
CSC	Color Space Conversion
FS	Frame Start
FE	Frame End
ISP	Image Signal Processing

2 Overview

The i.MX 8M series MIPI capture system supports incoming data from any MIPI CSI-2 compliant camera device and captures and packs this data into memory. It has the following main submodules:

- MIPI CSI-2 Rx subsystem
  - MIPI CSI-2 Rx DPHY
  - MIPI CSI-2 Rx controller
- Imaging subsystem
  - CSI Bridge for i.MX 8MQ and i.MX 8MM
  - ISI for i.MX 8MN and i.MX 8MP

Note: *Besides the ISI block, image sensor data can also be handled by the embedded ISP block on i.MX 8MP. This document does not describe the 8MP ISP block. For more information on how to port a new camera with 8MP ISP, refer to User Guide IMX8MPCSPUG.*

3 MIPI CSI-2 Rx subsystem

This section introduces the i.MX 8M series MIPI CSI-2 Rx subsystem with the CSI-2 Rx DPHY and host controller. The CSI-2 Rx DPHY and host controller are digital cores that implement all protocol functions defined in the MIPI CSI-2 specifications, providing an interface between the host processor and a MIPI CSI-2 compliant camera sensor.

In i.MX 8MM and i.MX 8MN, one instance of this subsystem is connected to the Imaging subsystem that handles the sensor/image input and process for all input imaging devices. There are two instances of the MIPI CSI-2 Rx subsystem implemented in i.MX 8MQ and i.MX 8MP. This document specifies a single instance of the CSI-2 Rx subsystem.

i.MX 8MM/i.MX 8MN/i.MX 8MP share the CSI-2 Rx DPHY and host controller IP while i.MX 8MQ uses a different IP.

3.1 i.MX 8MQ MIPI CSI-2 Rx subsystem

This section introduces the MIPI CSI-2 Rx subsystem in i.MX 8MQ.

Figure 1 shows the block diagram for the MIPI CSI-2 Rx subsystem in i.MX 8MQ.

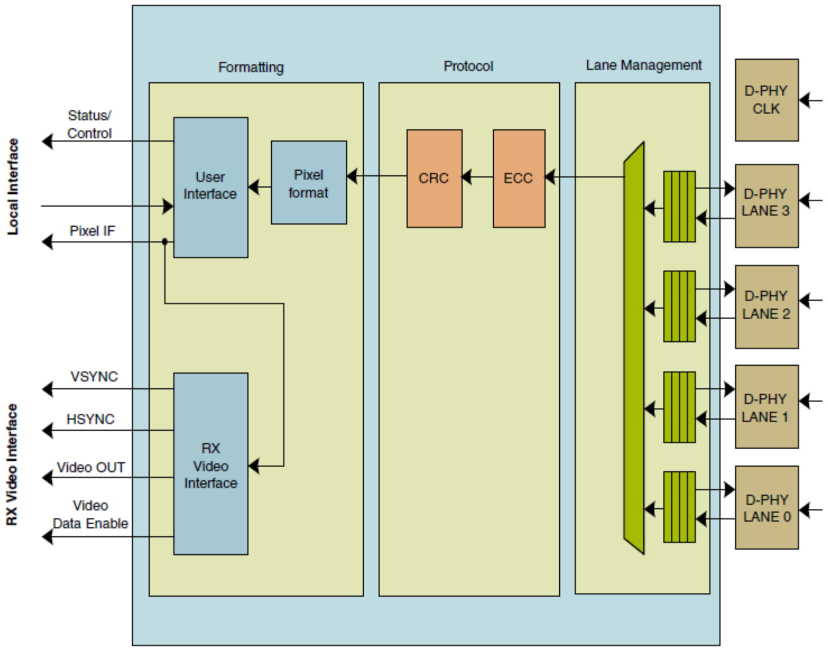

Figure 1. i.MX 8MQ MIPI CSI-2 Rx block diagram

3.1.1 Features

The i.MX 8MQ MIPI CSI-2 Rx subsystem has the following key features:

- Complies with MIPI D-PHY and CSI-2 Specification Version 1.1

- Consists of 1 Clock lane and up to 4 Data lanes
- Supports both high-speed and low-power modes
- Supports 80 Mbps to 1.5 Gbps data rate in high-speed mode per lane
- Supports 10 Mbps data rate in low-power mode
- Supports primary and secondary image format
  - YUV420, YUV420 (Legacy), YUV420 (CSPS), YUV422 of 8-bits and 10-bits
  - RGB565, RGB666, RGB888
  - RAW6, RAW7, RAW8, RAW10, RAW12, RAW14
  - All User-Defined Byte-based Data packet
- Support video data output with Hsync and Vsync.

3.1.2 Configuration port

The i.MX 8MQ CSI-2 Rx controller core has configuration ports that can control the configuration of the core in real time. The ports are listed and described in Table 2. When the CSR module is part of the CSI-2 Rx controller, the cfg_* ports are removed from the top level and connected to ports on the CSR module, making them accessible through the APB interface.

Table 2. i.MX 8MQ CSI-2 Rx controller core configuration port

Port	Type	Description
cfg_num_lanes[1:0]	Input	Sets the number of active lanes that are to be used for receiving MIPI data. [0] –1 Lane [1] – 2 Lanes [2] – 3 Lanes [3] – 4 Lanes
cfg_disable_data_lanes[3:0]	Input	Setting bits to a‘1’ value causes the DPHY Enable signal to deassert. [0] – data lane 0 [1] – data lane 1 [2] – data lane 2 [3] – data lane 3

3.1.3 Clocking constraint

The i.MX 8MQ CSI-2 Rx controller core requires two clocks for proper operation of clk_ui and clk. The clk_ui clock is the clock that the user interface is synchronous to, with all inputs and outputs referenced to the rising edge of clk_ui. The clk input is the clock that the CSI-2 Rx controller core uses to process data received from the D-PHY.

Table 3 describes the clock port and the bandwidth requirement.

Table 3. i.MX 8MQ CSI-2 Rx controller clock

Clock name	Description
clk (MIPI_CSIx_CORE_CLK)	Rx controller Core Clock (Byte Clock). `clk` must be greater than or equal to the received byte clock (RxByteClkHS%) coming from the data lane 0, 1, 2, 3 of Rx DPHY.
clk_ui (MIPI_CSIx_PHY_REF_CLK)	User Interface Clock (Pixel Clock). `clk_ui` must be greater than or equal to the incoming data from the CSI-2 MIPI interface.

Table 3. i.MX 8MQ CSI-2 Rx controller clock*...continued*

Clock name	Description
clk_esc (MIPI_CSIx_ESC_ CLK)	Rx Escape Clock. It must be the same escape clock that the Rx DPHY receives.

The minimum frequency for `clk_ui` must be high enough to keep up with the incoming data rate of the CSI-2 MIPI interface. It is important to realize that the user interface is pixel-based and therefore may be wider than the incoming byte data from the MIPI Rx DPHY. As an example, if RGB888 data is being received, the user interface data width will be 24 bits (1 pixel per clk_ui rising edge and each pixel is 24 bits). As an example, let the MIPI interface run at 1000 Mbps with a single lane. The data rate into the CSI-2 Rx controller is 125 MBps, so `clk` must run at 125 MHz or higher. Since the user interface is 3 bytes wide, `clk_ui` must run at 125MHz/3 or higher to keep up with the incoming data.

3.1.4 Configuration procedure

The reset and initialization procedure for the i.MX 8MQ MIPI CSI-2 Rx subsystem is as follows:

1. Assert all resets.
2. Wait until clocks to the Rx controller are stable and ensure that the DPHY interface is idle (Stop State on all lanes)
3. Deassert reset to CSR, `pclk_reset_n`, if CSR is included with the controller. If there is no CSR, go to step 4.
4. Program CSR registers if CSR is present. If CSR is not present, set values to all configuration ports (`cfg_* inputs`) to appropriate values.
5. Number of data lanes (register `CSI2RX_CFG_NUM_LANES` and `CSI2RX_CFG_DISABLE_DATA_LANES`)
6. Interrupt masks (register `CSI2RX_IRQ_MASK`)
7. PHY settings, such as `hs_settle` (register field `PRG_RXHS_SETTLE`, see Section 3.1.5)
8. Deassert all remaining resets.
9. i.MX 8MQ CSI-2 Rx controller is ready for use by the next rising edge of `clk_ui`.

3.1.5 DPHY high-speed settle timer

Table 4 and Table 5 describe how to calculate the value for `PRG_RXHS_SETTLE [5:0]`.

Table 4. High-speed settle timer range

	Min (ns)	Max (ns)
$T_{HS\text{-}SETTLE}$	85 ns + 6*UI	145 ns + 10*UI

Table 5. PRG_RXHS_SETTLE setting for different data rates

Data rate	Min (ns)	Max(ns)	PRG_RXHS_SETTLE[5:0]	$T_{HS\text{-}SETTLE}$ (ns)
1.5 Gbps	89	151.67	000110	106.06
1.0 Gbps	91	155	000110	106.06
500 Mbps	97	165	000110	106.06
250 Mbps	109	185	001000	136.36
80 Mbps	160	270	001011	181.82

1. The calculations assume that the frequency of RxClkInEsc is 66 MHz (Tperiod=15.15 ns)
2. $T_{HS\text{-}SETTLE}$= (PRG_RXHS_SETTLE + 1) * (Tperiod of RxClkInEsc)

Table 5. PRG_RXHS_SETTLE setting for different data rates...*continued*

Data rate	Min (ns)	Max(ns)	PRG_RXHS_SETTLE[5:0]	$T_{HS-SETTLE}$ (ns)
3. There is a calculation inaccuracy up to 1 cycle of RxClkInEsc, so it is always recommended to choose a programming value away from the limits.				

3.1.6 Register definition

This section lists some register fields that the user must pay attention to for porting a new camera. For more registers, refer to the i.MX 8MQ Reference Manual.

The base address for i.MX 8MQ MIPI CSI-2 controller is:

1. CSI-2 Port 1: 30A7_0000h
2. CSI-2 Port 2: 30B6_0000h
   - `CSI2RX_CFG_NUM_LANES` register: 100h offset

Table 6. **CSI2RX_CFG_NUM_LANES** register

Field	Description
1-0 csi2rx_cfg_num_lanes	Sets the number of active lanes that are to be used for receiving data. 00b - 1 Lane 01b - 2 Lane 10b - 3 Lane 11b - 4 Lane

- `CSI2RX_CFG_DISABLE_DATA_LANES` register: 104h offset

Table 7. **CSI2RX_CFG_DISABLE_DATA_LANES** register

Field	Description
3-0 csi2rx_cfg_disable_data_lanes	Setting bits to a ‘1’ value causes the DPHY Enable signal to deassert. 0001b - Data Lane 0 0010b - Data Lane 1 0100b - Data Lane 2 1000b - Data Lane 3

- `CSI2RX_IRQ_STATUS` register: 10Ch offset

Table 8. **CSI2RX_IRQ_STATUS** register

Field	Description
8-0 csi2rx_irq_status	CSI2 RX IRQ status [0] – crc error [1] – one-bit ecc error [2] – two-bit ecc error [3] – ULPS status change [4] – DPHY ErrSotHS has occurred [5] – DPHY ErrSotSync_HS has occurred [6] – DPHY ErrEsc has occurred [7] – DPHY ErrSyncEsc has occurred [8] – DPHY ErrControl has occurred

- `CSI2RX_IRQ_MASK` register: 110h offset

Table 9. CSI2RX_IRQ_MASK register

Field	Description
8-0 csi2rx_irq_mask	CSI2 RX IRQ Mask setting Each bit in IRQ_MASK corresponds to each bit in IRQ_STATUS. Setting a bit in IRQ_MASK to 1 will mask the corresponding bit in IRQ_STATUS from causing irq_out to assert.

- IOMUXC_GPR_GPR34 register for CSI-2 Port 1: 3034_0088h
  IOMUXC_GPR_GPR41 register for CSI-2 Port 2: 3034_00A4h

Table 10. IOMUXC_GPR_GPR34/41 register

Field	Description
7–2 CSI2_x_S_PRG_ RXHS_SETTLE	MIPI CSI2_x D-PHY program T_HS_SETTLE bits. HS-RX waits for Time-out T_HS_SETTLE in order to neglect transition effects. See Section 3.1.5, “DPHY High-Speed Settle Timer”.

3.2 i.MX 8MM/i.MX 8MN/i.MX 8MP MIPI CSI-2 Rx subsystem

This section introduces the MIPI CSI-2 Rx subsystem in i.MX 8MM, i.MX 8 MN and i.MX 8MP.

Figure 2 shows the block diagram for the MIPI CSI-2 Rx subsystem in i.MX 8MM/i.MX 8MN/i.MX 8MP. MIPI CSI-2 Rx controller (CSIS V3.6.3) works with the MIPI Rx DPHY module to connect camera to the host processor.

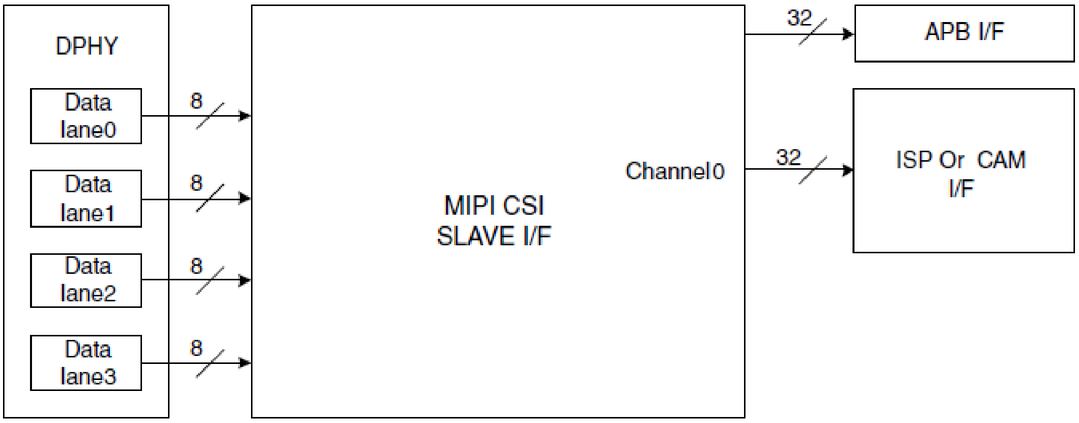

Figure 2. CSIS V3.6.3 block diagram

Figure 3 shows the block diagram for the CSIS V3.6.3 Slave Module in i.MX 8MM/i.MX 8MN/i.MX 8MP.

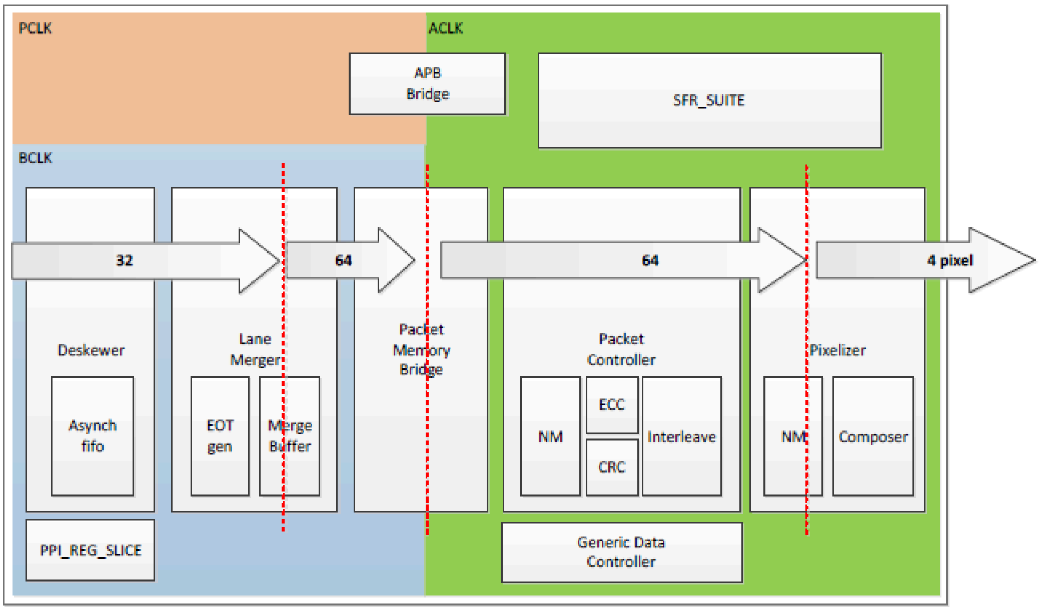

Figure 3. CSIS V3.6.3 slave module block diagram

3.2.1 Features

The i.MX 8MM/i.MX 8MN/i.MX 8MP MIPI CSI-2 Rx subsystem has the following key features:

- Complies with MIPI D-PHY Specification Version 1.2
- Complies with MIPI CSI-2 Specification Version 1.3
- Consists of 1 Clock lane and up to 4 Data lanes
- Supports both high-speed and low-power modes
- Supports 80 Mbps to 1.5 Gbps data rate in high-speed mode per lane
- Supports 10 Mbps data rate in low-power mode
- Supports primary and secondary image format
  - YUV420, YUV420 (Legacy), YUV420 (CSPS), YUV422 of 8-bits and 10-bits
  - RGB565, RGB666, RGB888
  - RAW6, RAW7, RAW8, RAW10, RAW12, RAW14
  - All User Defined Byte-based Data packet
- Support video data output with Hsync and Vsync.
- Pixel clock can be gated when no ppi data is coming.

3.2.2 ISP/CAM interface

CSIS V3.6.3 ISP or CAM interface has output signals which are PIX_CLK, VVALID, HVALID, BVALID, and DATA (described in Section 3.2.2). All signals are synchronized with the rising edge of `PIX_CLK`.

Table 11. CSIS V3.6.3 ISP/CAM interface port

Port	Type	Description
PIX_CLK	Output	Pixel clock. This signal is generated from I_ACLK.
VVALID	Output	Vertical sync signal. This signal is set during the whole frame.

Table 11. CSIS V3.6.3 ISP/CAM interface port*...continued*

Port	Type	Description
HVALID	Output	Horizontal sync signal. This signal is set during each line.
BVALID	Output	Byte valid signal. This signal indicates the valid number of bytes and is only valid in the Use defined type and Parallel mode.
DATA	Output	Pixel data bus. The pixel is aligned based on its data type.

3.2.3 Clocking constraint

MIPI CSI has 3 clock sources: I_RxByteClkHs%, I_ACLK and I_PCLK.

Table 12 describes the clock port and the bandwidth requirement.

Table 12. CSIS V3.6.3 Clock

Clock name	Description
I_RxByteClkHs%	Byte Clock. This clock comes from data lane 0,1,2,3 of Rx DPHY.
I_ACLK	Pixel Clock. This clock is an external clock for pixel clock what is for using ISP/CAM interface.
I_PCLK	APB Clock. This clock is system clock generated by general processor PLL.

The relationship between input and output bandwidth is that the output bandwidth should be faster than input bandwidth. There is an equation of previous relationship:

RX_BYTE_CLK_HS x NUMBER_OF_DATA_LANE x 8BITS ≤ PIXEL_CLOCK x BITWIDTH_OF_IMAGE_FORMAT x NUMBER_OF_PIXEL_PER_CLOCK

3.2.4 Configuration procedure

The reset and initialization procedure for the i.MX 8MM/i.MX 8MN/i.MX 8MP MIPI CSI-2 Rx subsystem is as follows:

1. Assert SW reset. It is deasserted automatically after the software reset is done.
2. Configure DPHY:
   a. DPHY slave control (register `MIPI_CSIx_DPHY_SLAVE_CTRL_LOW`)
   b. DPHY master and slave control (register `MIPI_CSIx_DPHY_MASTER_SLAVE_CTRL_LOW/HIGH`)
   c. DPHY common control, such as enabling data lane 0/1/2/3, setting HS settle timer (`MIPI_CSIx_DPHY_COMMON_CTRL`)
3. Configure CSIS:
   a. Interrupt mask (register `MIPI_CSIx_INTERRUPT_MASK_0`)
   b. CAM/ISP interface settings, such as image data type, vertical/horizontal image resolution (register `MIPI_CSIx_ISP_CONFIG0, MIPI_CSIx_ISP_RESOLUTION0`, and `MIPI_CSIx_ISP_SYNC0`)
   c. CSIS clock control (register `MIPI_CSIx_CSIS_CLOCK_CTRL`)
   d. CSIS common control, such as the number of data lanes (register `MIPI_CSIx_CSIS_COMMON_CTRL`)
4. Configure Gasket (only for i.MX 8 MN and i.MX 8MP):
   a. Gasket control, such as data type, interlace mode (register `GPR_GASKET_x_CTRL`)

b. Gasket video size setting (register GPR_GASKET_x_HSIZE and GPR_GASKET_x_VSIZE)
   Note: *In i.MX 8 MN and i.MX 8MP, there is a block named Gasket between the MIPI CSI-2 Rx subsystem and Imaging subsystem, it contains general-purpose registers to control varied features of the associated peripherals and must be configured properly depending on the use case.*

5. Enable Rx controller, DPHY, and Gasket.

3.2.5 DPHY high-speed settle timer

RX_HS_SETTLE [7:0] must be set according to the MIPI data rate on each lane (see Table 13).

Table 13. RX_HS_SETTLE setting for different data rates

Data rate (Mbit/s)	RX_HS_SETTLE [7:0]
1500	33
1490~1450	32
1440~1410	31
1400~1360	30
1350~1320	29
1310~1270	28
1260~1230	27
1220~1180	26
1170~1130	25
1120~1090	24
1080~1040	23
1030~1000	22
990~950	21
940~910	20
900~860	19
850~820	18
810~770	17
760~730	16
720~680	15
670~640	14
630~590	13
580~550	12
540~500	11
490~460	10
450~410	9
400~370	8
360~320	7
310~280	6
270~230	5

Table 13. RX_HS_SETTLE setting for different data rates*...continued*

Data rate (Mbit/s)	RX_HS_SETTLE [7:0]
220~190	4
180~140	3
130~100	2
90~80	1

3.2.6 Register definition

This section lists some register fields that the user must pay attention to for porting a new camera. For more registers, please refer to the i.MX 8MM, i.MX 8MN, or i.MX 8MP Reference Manual.

The base address for MIPI CSI-2 controller is:

1. i.MX 8MN/8MM CSI-2 Port 1: 32E3_0000h
2. i.MX 8MP CSI-2 Port 1: 32E4_0000h
3. i.MX 8MP CSI-2 Port 2: 32E5_0000h
   - `MIPI_CSIx_CSIS_COMMON_CTRL` register: 4h offset

Table 14. **MIPI_CSIx_CSIS_COMMON_CTRL register**

Field	Description
9-8 LANE_NUMBER	Number of data lanes 00 1 data lane 01 2 data lane 10 3 data lane 11 4 data lane

- `MIPI_CSIx_INTERRUPT_MASK_0`: 10h offset

Table 15. **MIPI_CSIx_INTERRUPT_MASK_0**

Field	Description
24 MSK_ FRAMESTART	FS packet is received, CH0 0 Disable (masked) 1 Enable (unmasked)
20 MSK_ FRAMEEND	FE packet is received, CH0 0 Disable (masked) 1 Enable (unmasked)
19-16 MSK_ERR_ SOT_HS	Start of transmission error [Lane3, Lane2, Lane1, Lane0] 0 Disable (masked) 1 Enable (unmasked)
12 MSK_ERR_ LOST_FS	Lost of Frame Start packet, CH0 0 Disable (masked) 1 Enable (unmasked)
8 MSK_ERR_ LOST_FE	Lost of Frame End packet, CH0 0 Disable (masked) 1 Enable (unmasked)
4 MSK_ERR_	Image FIFO overflow interrupt 0 Disable (masked)

Table 15. MIPI_CSIx_INTERRUPT_MASK_0 *...continued*

Field	Description
OVER	1 Enable (unmasked)
3 MSK_ERR_ WRONG_CFG	Wrong configuration 0 Disable (masked) 1 Enable (unmasked)
2 MSK_ERR_ECC	ECC error 0 Disable (masked) 1 Enable (unmasked)
1 MSK_ERR_CRC	CRC error 0 Disable (masked) 1 Enable (unmasked)
0 MSK_ERR_ID	Unknown ID error 0 Disable (masked) 1 Enable (unmasked)

- MIPI_CSIx_INTERRUPT_SOURCE_0: 14h offset

Table 16. MIPI_CSIx_INTERRUPT_SOURCE_0

Field	Description
24 FRAME_START	FS packet is received, CH0
20 FRAME_END	FE packet is received, CH0
19-16 ERR_SOT_HS	Start of transmission error [Lane3, Lane2, Lane1, Lane0] 0 Disable (masked) 1 Enable (unmasked)
12 ERR_LOST_FS	Indication of lost of Frame Start packet, CH0 This field is set to '1' when the image data is received without the frame start packet.
8 ERR_LOST_FE	Indication of lost of Frame End packet, CH0 This field is set to '1' when the image data is received without the frame end packet.
4 ERR_OVER	Overflow is caused in image FIFO The outer bandwidth must be faster than the inner bandwidth. But the image FIFO can be overflow because of the user’s fault. There are 2 ways for preventing overflow. • Tune the output pixel clock faster than the current. • Tune the input byte clock slower than the current. When this interrupt is generated: • Turn off the camera • Assert software reset, if you do not assert software reset, MIPI CSIS cannot receive any more data. • Tune the clock frequency and reconfigure all related registers. MIPI CSIS module is ready for operating.
3 ERR_WRONG_CFG	Wrong configuration The received packet is not allocated to the ISP or CAM interface channel due to a wrong data type or virtual channel id.
2 ERR_ECC	ECC error

Table 16. MIPI_CSIx_INTERRUPT_SOURCE_0*...continued*

Field	Description
1 ERR_CRC	CRC error
0 ERR_ID	Unknown ID error

- MIPI_CSIx_DPHY_STATUS register: 20h offset

Table 17. MIPI_CSIx_DPHY_STATUS

Field	Description
11–8 ULPSDAT	Data lane [3:0] is in ULPS [7] : data lane 3 [6] : data lane 2 [5] : data lane 1 [4] : data lane 0 0 Not ULPS 1 ULPS
7–4 STOPSTATEDAT	Data lane [3:0] is in Stop State [7] : data lane 3 [6] : data lane 2 [5] : data lane 1 [4] : data lane 0 0 Not Stop state 1 Stop state
1 ULPSCLK	0 Not ULPS 1 ULPS
0 STOPSTATECLK	Clock lane is in Stop State 0 Not Stop state 1 Stop state

- MIPI_CSIx_DPHY_COMMON_CTRL register: 24h offset

Table 18. MIPI_CSIx_DPHY_COMMON_CTRL register

Field	Description
31-24 HSSETTLE[7:0]	HS-RX settle time control Slave Clock Lane Control for Ths-settle. See Section 3.2.5.
4-1 ENABLE_DAT[3:0]	DPHY data lane enable [3] : data lane 3 [2] : data lane 2 [1] : data lane 1 [0] : data lane 0 0 Disable 1 Enable
0 ENABLE_CLK	DPHY clock lane enable 0 Disable

Table 18. `MIPI_CSIx_DPHY_COMMON_CTRL` register*...continued*

Field	Description
	1 Enable

- `MIPI_CSIx_ISP_CONFIG0`: 40h offset

Table 19. `MIPI_CSIx_ISP_CONFIG0`

Field	Description
13-12 `PIXEL_MODE`	Pixel mode selection 01 : dual pixel mode (YUV422) 00 : single pixel mode
7-2 `DATAFORMAT[5:0]`	Image Data Format 0x18 - YUV420 (8bit) 0x19 - YUV420 (10bit) 0x1A - YUV420 (8bit legacy) 0x1C - YUV420 (8bit CSPS) 0x1D - YUV420 (10bit CSPS) 0x1E - YUV422 (8bit) 0x1F - YUV422 (10bit) 0x20 - (NOT SUPPORTED) RGB444 0x21 - (NOT SUPPORTED) RGB555 0x22 - RGB565 0x23 - RGB666 0x24 - RGB888 0x28 - RAW6 0x29 - RAW7 0x2A - RAW8 0x2B - RAW10 0x2C - RAW12 0x2D - RAW14 0x30 - User defined 1 0x31 - User defined 2 0x32 - User defined 3 0x33 - User defined 4 0x34 - User defined 5 0x35 - User defined 6 0x36 - User defined 7 0x37 - User defined 8 NOTE: Not described types are ignored.

- `MIPI_CSIx_ISP_RESOLUTION0:` 44h offset

Table 20. `MIPI_CSIx_ISP_RESOLUTION0`

Field	Description
31-16 `VRESOL`	Vertical Image resolution Input boundary: 0x0001 ~ 0xFFFF
15-0 `HRESOL`	Horizontal Image resolution HRESOL value must comply with the terms of the standard spec and it must be a multiple of four. Input boundary: 0x0004 ~ 0xFFFC

The base address for Gasket is:

a. Gasket 0 for i.MX 8MN CSI-2 Port 1: 32E2_8060h
b. Gasket 0 for i.MX 8MP CSI-2 Port 1: 32EC_0060h
c. Gasket 1 for i.MX 8MP CSI-2 Port 2: 32EC_0090h

- `GASKET_x_CTRL`: 0h offset

Table 21. **`GASKET_x_CTRL`**

Field	Description
13-8 `GASKET_x_DATA_TYPE`	Gasket x data type 011000b - YUV420 8-bit 011001b - YUV420 10-bit 011010b - Legacy YUV420 8-bit 011100b - YUV420 8-bit(Chroma Shifted Pixel Sampling) 011101b - YUV420 10-bit(Chroma Shifted Pixel Sampling) 011110b - YUV422 8-bit 011111b - YUV422 10-bit 100010b - RGB565 100011b - RGB666 100100b - RGB888 101000b - RAW6 101001b - RAW7 101010b - RAW8 101011b - RAW10 101100b - RAW12 101101b - RAW14
7-6 `GASKET_x_INTER_MODE`	Gasket x interlace mode 00b - not interlaced 01b - interlaced left 10b - interlaced right 11b - reserved
1 `GASKET_x_DOUBLE_COMP`	Gasket x double component enable 0b - Gasket x input single component per pixel clock for YUV422 1b - Gasket x input double component per pixel clock for YUV422

Note: *The `GASKET_x_INTER_MODE` field must be configured to transfer interlaced image data to ISI in which deinterlace feature is supported.*

- `GASKET_x_HSIZE`: 4h offset

Table 22. **`GASKET_x_HSIZE`**

Field	Description
31-0 `GASKET_x_HSIZE`	Gasket x video Horizontal size (count in pixel)

- `GASKET_x_VSIZE`: 8h offset

Table 23. **`GASKET_x_VSIZE`**

Field	Description
31-0	Gasket x video Vertical size (count in line)

Table 23. GASKET_x_VSIZE

Field	Description
GASKET_x_VSIZE	

- GASKET_x_ISI_PIXEL_CNT: 1Ch offset

Table 24. GASKET_x_ISI_PIXEL_CNT

Field	Description
31-0 GASKET_x_ISI_PIXEL_CNT	Gasket x output to ISI pixel count status

- GASKET_x_ISI_LINE_CNT: 20h offset

Table 25. GASKET_x_ISI_LINE_CNT

Field	Description
GASKET_x_ISI_LINE_CNT	Gasket x output to ISI line count status

4 Imaging subsystem

The i.MX 8M series imaging subsystem supports incoming data from MIPI CSI-2 RX subsystem and captures and packs this data into memory.

4.1 CSI bridge

CSI Bridge is implemented in i.MX 8MQ and i.MX 8MM.

Figure 4 shows the block diagram for CSI bridge.

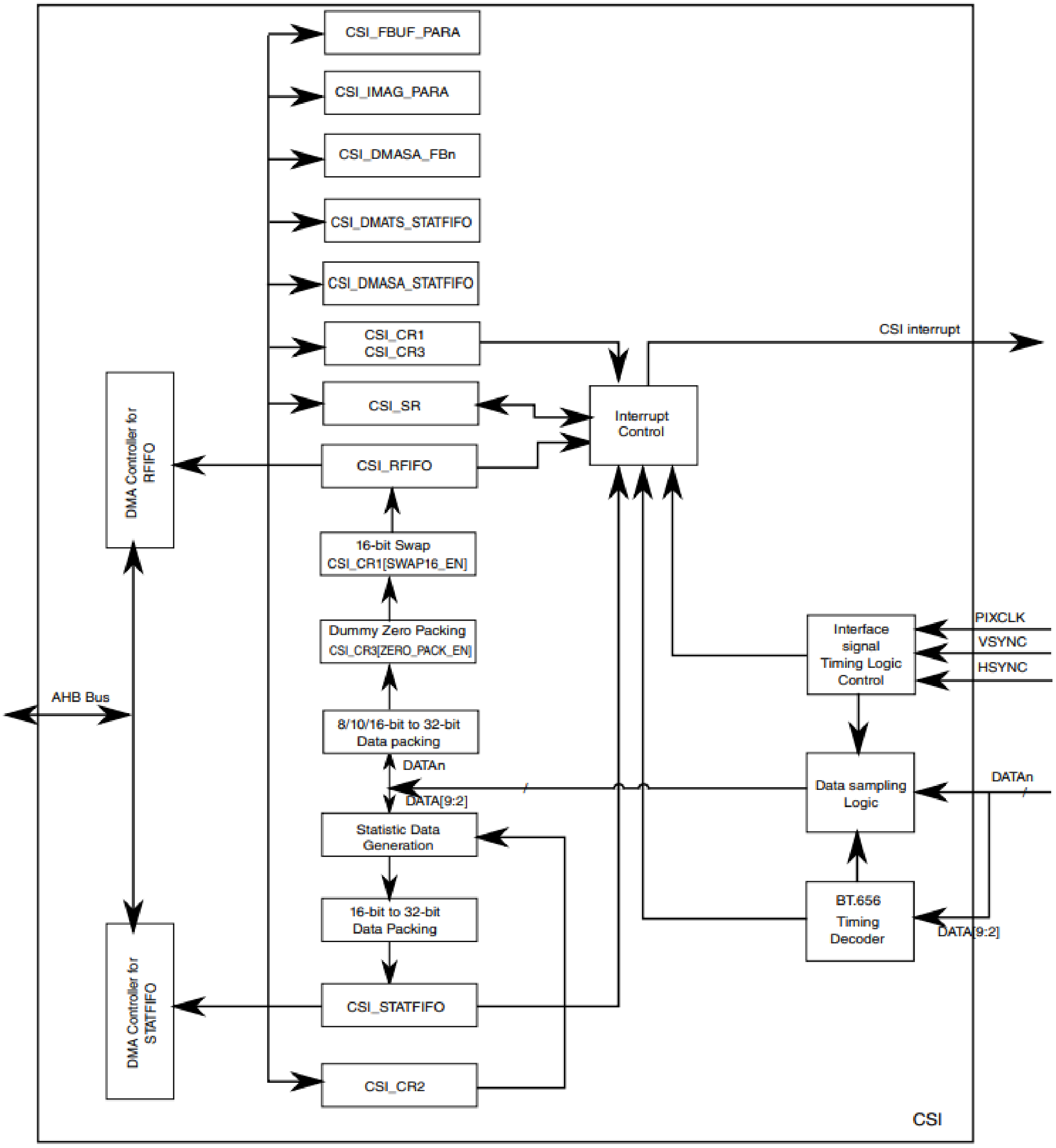

Figure 4. CSI bridge block diagram

Note: *This section lists the features and configuration procedure for the CSI Bridge. Refer to the i.MX 8MQ or i.MX 8MM Reference Manual for more details of this IP, such as the operation principles, interrupt generation, data packing style and so on.*

4.1.1 Features

CSI Bridge IP has the following key features:

- Configurable interface logic to support most commonly available CMOS sensors
- Support 8-bit / 16-bit / 24-bit data ports for YCbCr, YUV, or RGB data input.
  - YUV420, YUV420 (Legacy), YUV420 (CSPS), YUV422 of 8-bits and 10-bits
  - RGB565, RGB666, RGB888
  - RAW6, RAW7, RAW8, RAW10, RAW12, RAW14
  - All User Defined Byte-based Data packet
- Full control of 8-bit/pixel, 10-bit/pixel, or 16-bit / pixel data format to 64-bit receive FIFO packing.
- 256 x 64 FIFO to store received image pixel data.
- Receive FIFO overrun protection mechanism.
- Embedded DMA controllers to transfer data from receive FIFO or statistic FIFO through AHB bus.
- Support 2D (data is chosen as x/y coordinate) DMA transfer from the receive FIFO to the frame buffers
- Support double buffering of two frames in the external memory.
- Single interrupt source to interrupt controller from maskable interrupt sources: Start of Frame, End of Frame, Change of Field, FIFO full, FIFO overrun, DMA transfer done, and AHB bus response error.
- Configurable master clock frequency output to the sensor.
- Statistic data generation for Auto Exposure (AE) and Auto White Balance (AWB) control of the camera (only for Bayer data and 8-bit/pixel format).

4.1.2 Configuration procedure

The initialization procedure for CSI Bridge is as follows:

- Configure the CSI to source from the MIPI camera by setting `CSI_CR18[DATA_FROM_MIPI]` to 1.
- Configure the image width and height in the `CSI_IMAG_PARA` register.
- Configure associated fields according to the sensor data type:
  - `MIPI_DATA_FORMAT, MIPI_YU_SWAP, MIPI_DOUBLE_CMPNT, RGB888A_FORMAT_SEL` and `PARALLEL24_EN` fields in the `CSI_CR18` register
  - `TWO_8BIT_SENSOR` and `ZERO_PACK_EN` fields in the `CSI_CR3` register
- Configure embedded DMA controller:
  - Asynchronous clear RxFIFO by writing ‘1’ to the `CSI_CR1[CLR_RXFIFO]` bit and writing ‘0’ to the `CSI_CR1[FCC]` bit.
  - Reflash the DMA controller for RxFIFO by writing ‘1’ to the `CSI_CR3[DMA_REFLASH_RFF]` bit.
  - Reset frame counter by writing ‘1’ to the `CSI_CR3[FRMCNT_RST]` bit.
  - Configure the base address-switching method in the `CSI_CR18[BASEADDR_SWITCH_SEL/EN]` field.
  - Configure the burst type of transfer in the `CSI_CR2[DMA_BURST_TYPE_RFF]` field.
  - Enable DMA Request for RxFIFO by writing ‘1’ to the `CSI_CR3[DMA_REQ_EN_RFF]` bit.
  - Configure the RxFIFO full level in the `CSI_CR3[RxFF_LEVEL]` field.
  - Configure the memory address where the image data has to be stored in the `CSI_DMASA_FB1` and `CSI_DMASA_FB2` registers.
- Configure interrupts:
  - Start or End of Frame Interrupt: the `SOF_INTEN` and `EOF_INT_EN` fields in the CSI_CR1 register
  - RxFIFO Overrun Interrupt: the `RF_OR_INTEN` field in the `CSI_CR1` register
  - Frame buffer 1/2 DMA Transfer Done Interrupt: the `FB1_DMA_DONE_INTEN` and `FB2_DMA_DONE_INTEN` fields in the `CSI_CR1` register
- Enable CSI Bridge by writing ‘1’ to the `CSI_CR18[CSI_ENABLE]` bit.
- Monitor the status of CSI Bridge through the `CSI_SR` register for the software to take appropriate action.

4.1.3 Register definition

This section lists some register fields that the user must pay attention to for porting a new camera. For more registers, refer to the i.MX 8MQ or i.MX 8MM Reference Manual.

The base address for CSI Bridge is:

1. i.MX 8MM CSI-2 Port 1: 32E2_0000h
2. i.MX 8MQ CSI-2 Port 1: 30A9_0000h
3. i.MX 8MQ CSI-2 Port 2: 30B8_0000h

- CSI_CR1 register: 0h offset

Table 26. CSI_CR1 register

Field	Description
29 EOF_INT_EN	End-of-Frame Interrupt Enable. 0 Disable 1 Enable
24 RF_OR_INTEN	RxFIFO Overrun Interrupt Enable. 0 Disable 1 Enable
20 FB2_DMA_DONE_INTEN	Frame Buffer2 DMA Transfer Done Interrupt Enable. 0 Disable 1 Enable
19 FB1_DMA_DONE_INTEN	Frame Buffer1 DMA Transfer Done Interrupt Enable. 0 Disable 1 Enable
16 SOF_INTEN	Start Of Frame (SOF) Interrupt Enable. 0 Disable 1 Enable
8 FCC	FIFO Clear Control. This bit determines how the RXFIFO and STATFIFO are cleared. When Synchronous FIFO clear is selected, the RXFIFO are cleared on every SOF. FIFOs restarts immediately after reset. For information on the operation when Asynchronous FIFO clear is selected, refer to the descriptions for the CLR_RXFIFO. 0 Asynchronous FIFO clear is selected. 1 Synchronous FIFO clear is selected.
5 CLR_RXFIFO	Asynchronous RXFIFO Clear. This bit clears the RXFIFO and works only in async FIFO clear mode-that is, FCC = 0. Otherwise this bit is ignored. Writing 1 clears the RXFIFO immediately, RXFIFO restarts immediately after that. The bit is restored to 0 automatically after finish. Normally reads 0.

- CSI_CR3 register: 8h offset

Table 27. SI_CR3 register

Field	Description
31–16 FRMCNT	Frame Counter. It is a 16-bit Frame Counter (Wraps around automatically after reaching the maximum)
15 FRMCNT_RST	Frame Count Reset. Resets the Frame Counter. (Cleared automatically after reset is done) 0 Do not reset 1 Reset frame counter immediately

Table 27. SI_CR3 register *...continued*

Field	Description
14 DMA_REFLASH_RFF	Reflash DMA controller for RxFIFO. This bit reflash the embedded DMA controller for RxFIFO. It should be reflashed before the embedded DMA controller starts to work. (Cleared automatically after reflashing is done) 0 No reflashing 1 Reflash the embedded DMA controller
12 DMA_REQ_EN_RFF	DMA Request Enable for RxFIFO. This bit enables the dma request from RxFIFO to the embedded DMA controller. 0 Disable the dma request 1 Enable the dma request
3 SENSOR_16BITS	16-bit Sensor Mode. This bit indicates one 16-bit sensor connected to the 16-bit data ports. This bit should be set if there is one 16-bit sensor connected. This bit should be configured before reflashing or restarting the embedded DMA controller. 0 Only one 8-bit sensor is connected. 1 One 16-bit sensor is connected.

- CSI_SR register: 18h offset

Table 28. CSI_SR register

Field	Description
28 BASEADDR_ CHHANGE_ ERROR	When using base address switching enable, this bit will be 1 when switching occur before DMA complete. This bit is clear by writing 1. When this interrupt happens, follow the steps listed below. 1. Deassert the CSI enable, CSIx_CSICR18 bit31, 2. Reflash the DMA, assert the CSIX_CSICR3 bit 14, 3. Assert the CSI enable, CSIx_CSICR18 bit31.
24 RF_OR_INT	RxFIFO Overrun Interrupt Status. Indicates the overflow status of the RxFIFO register. (Cleared by writing 1) 0 RXFIFO has not overflowed. 1 RXFIFO has overflowed.
20 DMA_TSF_ DONE_FB2	DMA Transfer Done in Frame Buffer2. Indicates that the DMA transfer from RxFIFO to Frame Buffer2 is completed. It can trigger an interrupt if the corresponding enable bit is set in CSICR1. This bit can be cleared by writing 1 or reflashing the RxFIFO dma controller in CSICR3. 0 DMA transfer is not completed. 1 DMA transfer is completed.
19 DMA_TSF_DONE_FB2	DMA Transfer Done in Frame Buffer1. Indicates that the DMA transfer from RxFIFO to Frame Buffer1 is completed. It can trigger an interrupt if the corresponding enable bit is set in CSICR1. This bit can be cleared by writing 1 or reflashing the RxFIFO dma controller in CSICR3. 0 DMA transfer is not completed. 1 DMA transfer is completed.
17 EOF_INT	End of Frame (EOF) Interrupt Status. Indicates when EOF is detected. (Cleared by writing 1) 0 EOF is not detected. 1 EOF is detected.
16 SOF_INT	Start of Frame Interrupt Status. Indicates when SOF is detected. (Cleared by writing 1) 0 SOF is not detected. 1 SOF is detected.

- CSI_IMAG_PARA register: 34h offset

Table 29. CSI_IMAG_PARA register

Field	Description
31–16 IMAGE_WIDTH	Image Width. Indicates the number of active pixel cycles per line. If the input data from the sensor is 8-bit/pixel format, the IMAGE_WIDTH should be a multiple of 8 pixels. If the input data from the sensor is 10-bit/pixel or 16-bit/pixel format, the IMAGE_WIDTH should be a multiple of 4 pixels.
15-0 IMAGE_HEIGHT	Image Height. Indicates how many pixels in a column of the image from the sensor.

- CSI_CR18 register: 48h offset

Table 30. CSI_CR18 register

Field	Description
30-25 MIPI_DATA_ FORMAT	Image Data Format Generic Short Packet: 0x08 ~ 0x0F Embedded 8-bit non Image: 0x12 YUV420 (8-bit) 0x18 YUV420 (10-bit) 0x19 YUV420 (8-bit legacy) 0x1A YUV420 (8-bit CSPS) 0x1C YUV420 (10-bit CSPS)0x1D YUV422 (8-bit) 0x1E YUV422 (10-bit) 0x1F RGB444 0x20 (Not support) RGB555 0x21 (Not support) RGB565 0x22 RGB666 0x23 RGB888 0x24 RAW6 0x28 RAW7 0x29 RAW8 0x2A RAW10 0x2B RAW12 0x2C RAW14 0x2D User defined 1 0x30 User defined 2 0x31 User defined 3 0x32 User defined 4 0x33 User defined 5 0x34 User defined 6 0x35 User defined 7 0x36 User defined 8 0x37
22 DATA_FROM_MIPI	0 Data from parallel sensor (Not supported on i.MX 8MM and i.MX 8MQ) 1 Data from MIPI
21 MIPI_YU_SWAP	It only works in MIPI CSI YUV422 double component mode.
20	Double component per clock cycle in YUV422 formats.

Table 30. CSI_CR18 register *...continued*

Field	Description
MIPI_DOUBLE_ CMPNT	0 Single component per clock cycle (half pixel per clock cycle) 1 Double component per clock cycle (a pixel per clock cycle)
19–18 MASK_OPTION	These bits used to choose the method to mask the CSI input. 00 Writing to memory (OCRAM or external DDR) from first completely frame, when using this option, the CSI_ENABLE should be 1. 01 Writing to memory when CSI_ENABLE is 1. 02 Writing to memory from second completely frame, when using this option, the CSI_ENABLE should be 1. 03 Writing to memory when data comes in, not matter the CSI_ENABLE is 1 or 0.

10
RGB888A_ FORMAT_SEL
	Output is 32-bit format.
	0 {8'h0, data[23:0]}
	1 {data[23:0], 8'h0}

9 BASEADDR_ CHANGE_ ERROR_IE	Base address change error interrupt enable signal. 0 Interrupt disabled 1 Interrupt enabled
5 BASEADDR_ SWITCH_SEL	CSI 2 base addresses switching method. When using this bit, BASEADDR_SWITCH_EN is 1. 0 Switching base address at the edge of the vsync 1 Switching base address at the edge of the first data of each frame
4 BASEADDR_ SWITCH_EN	When this bit is enabled, CSI DMA will switch the base address according to BASEADDR_ SWITCH_SEL rather than automatically by DMA completed.
3 PARALLEL24_EN	Enable bit for Parallel RGB888/YUV444 24bit input 0 Input is disabled 1 Input is enabled

4.2 Image sensing interface

ISI is implemented in i.MX 8MN and i.MX 8MP. It is a simple camera interface that supports image processing and transfer via a bus master interface.

Figure 5 shows the block diagram for ISI.

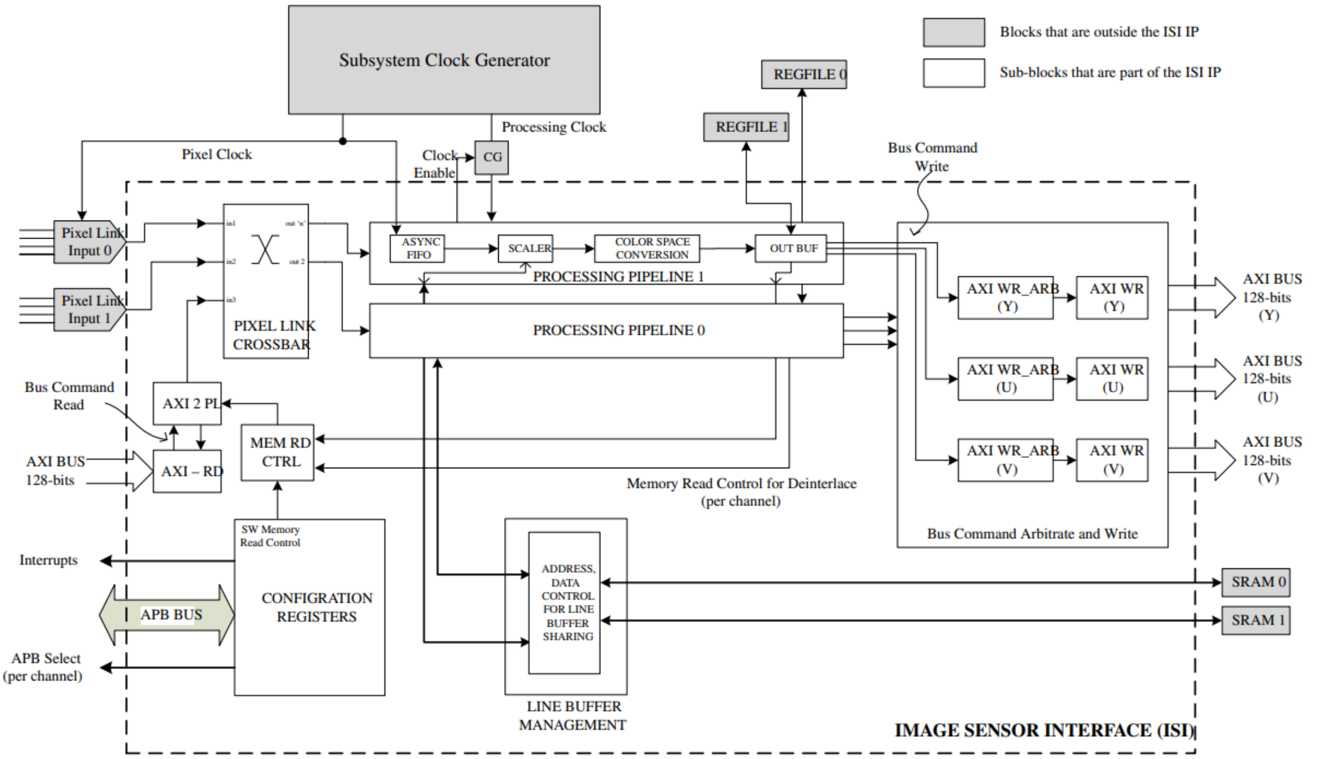

Figure 5. ISI block diagram

Note: *This section lists the features and configuration procedure for ISI. Refer to the i.MX 8MN or i.MX 8MP Reference Manual for more details of this IP, such as the design overview, functional description and so on.*

4.2.1 Features

ISI IP has the following key features:

- Supports up to 2K width on each channel for processed camera stream.
- High-resolution image handling
  - 4K width-processed camera stream can be supported by chaining two adjacent processing channels.
- Input sources supported:
  - 2 pixel link interfaces running in parallel that can interface to 2 camera sensors.
  - System Memory (AXI master; internally converted to Pixel Link Interface).
- Each processing pipeline or channel can be assigned to the same or different pixel input source.
- Stream Multiplexing features:
  - Duplicates input stream into multiple outputs.
  - Simple De-Interlacing methods supported for interlaced input sources:
    - Weaving
    - Line Doubling
    - Blending
  - One input source can be processed in more than one processing channels.
- Stream manipulation features:
  - Supported pixel formats when storing image into memory
    - YUV444, YUV422, YUV420 (8-bit, 10-bit, 12-bit) in planar or semi-planar formats
    - RGB565, RGB666, RGB888

    - RAW8, RAW10, RAW12, RAW16
    - Plus more formats listed in the description of FORMAT field in the `IMG_CTRL` register
  - Downscaling of input image via Decimation and Bilinear filtering
    - Decimation by 2, 4, or 8 supported
    - Bilinear filter further downscales by 1.0 to 2.0 (fractional downscaling)
  - Color space conversion
    - RGB, YUV, YCbCr
    - User-defined color space matrix-based conversion
  - Alpha channel insertion for RGB formats
    - Global alpha value
    - Separate rectangular region of interest (ROI) alpha value. ROI alpha values have higher priority than global alpha value. Up to 4 ROI non-overlapping rectangles supported.
  - Mirroring (Image Flip): Horizontal and Vertical flips supported
  - Frame awareness, Frame skipping
    - Clean frame start, shutdown based on HSYNC and VSYNC
    - Buffer overrun protection
    - Buffer under run deterministic behavior
- Stream output options
  - Input source is converted to and processed by the processing pipeline YUV or RGB.
  - Processed images are output from pipeline and stored into memory location specified by software.
  - Full line storage available at processing channel output before outputting data to AXI.
  - Dual buffered addresses used in ping pong fashion with active buffer status indication.
  - Line and Frame stored interrupt status to software to track progress of frame.
- Flow control
  - Panic indication to software and device to increase priority of its write transactions to avoid potential overflow in output buffers. Software can configure thresholds for panic indication.
  - When pixels are sourced from memory or input line buffers are flushed, software has the option to program the rate at which pixels will be sent out. By default, one pixel per clock is sent out.
  - Back pressure mechanism to stall the channel pipeline during line buffer flushing and sourcing image from memory, when AXI bus is unable accept data and the output buffers are low on storage.

Note: *Some ISI features rely on multi ISI processing pipelines (channels): all the stream multiplexing features except weaving de-interlacing, down scaling feature for YUV420 format and high-resolution image handling. There are two processing channels implemented in 8MP ISI, while only one processing channel implemented in 8 MN ISI.*

4.2.2 Configuration procedure

This section describes the suggested steps to configure the individual channels in ISI and handle interrupts for proper and efficient image processing. Every channel that must be configured must follow these suggested initialization steps for the correct operation of the module.

First, the general channel configuration is done as follows:

- Set the `CNHL_CTRL[CLK_EN]` bit to 1 to enable the clock to the channel. This is required to access other registers in the channel and also for image processing by channel.
- If the channel is supposed to be bypassed that is, No Down-scaling and Color Space Conversion (CSC) is needed; set the `CHNL_CTRL[CHNL_BYPASS]` bit to 1, else set it to 0. It causes the image coming into the channel to be directly stored in memory without any kind of processing. However, image flipping (horizontal

and/or vertical), image cropping, alpha insertion (RGB format only), and format conversion (YUV444 to YUV422 or YUV420) can still be done while the bypass is enabled.

- Select the source of the input image. Configure the source in the `CHNL_CTRL[SRC]` field.
  - For i.MX 8MP, the camera input can be any of the 2 Pixel Links (0 to 1). An additional pixel link, (Pixel Link 2), can be selected as a source. Pixel Link 2 always reads an image from a programmed location in memory.
  - For i.MX 8MN, only Pixel Link 0 is for streaming camera input. Pixel Link 1 is for reading the image from memory.
- Select the type of source for the channel between non-memory Pixel Links (image sourced from the camera sensor, SRC_TYPE = 0) and Memory Pixel Link (image sourced from memory, SRC_TYPE = 1). A memory Pixel Link sources the image from a programmed memory location only. It is useful for de-interlacing and processing images directly from memory. For channels that do not support sourcing images from memory, the source type is always a Non-memory Pixel Link. Configure the source type for the channel in the `CHNL_CTRL[SRC_TYPE]` field.
- Configure the image width and height in the `CHNL_IMG_CFG` register.
- Configure the ISI output image format in the `CHNL_IMG_CTRL[FORMAT]` field.
- Configure the memory address where the processed or bypassed data from the channel has to be stored. The image can be stored as 1-planar (all color components stored together), 2-planar (Y component stored in one memory address and UV stored in a different memory address), and 3-planar (each component stored in a different memory address).
  - For 1-planar addressing, configure the 32-bit address in `CHNL_OUT_BUF1_ADDR_Y` and `CHNL_OUT_BUF2_ADDR_Y` registers.
  - For 2-planar and 3-planar addressing, the `CHNL_OUT_BUF1_ADDR_U` and `CHNL_OUT_BUF2_ADDR_U` are additionally used.
  - For 3-planar addressing, the `CHNL_OUT_BUF1_ADDR_V` and `CHNL_OUT_BUF2_ADDR_V` are additionally used.

    Note: *RGB and RAW images are always stored as 1-planar while other YUV formats can be stored as 1/2/3-planar.*
- Configure the output buffer line pitch in the `CHNL_OUT_BUF_PITCH[LINE_PITCH]` register. The line pitch is the number of bytes between the start location of two lines. It is used to have line addresses start at an aligned boundary.

  Note: *The line pitch is the same for all image components irrespective of whether they are stored in 1-planar, 2-planar, or 3-planar. In 2 and 3-planar output, the U and V component have the same line pitch as the Y component. Ensure that the line pitch is programmed to be greater than the number of bytes in one line.*
- Toggle the address load bits that are, `CHNL_OUT_BUF_CTRL[LOAD_BUF1_ADDR/ LOAD_BUF2_ADDR]` to allow new configured addresses to be picked and used in the channel when it is enabled later.

Follow the steps detailed in appropriate sections from i.MX 8MP Reference Manual based on which functionality is to be used for the application.

- Refer to the 13.4.3.2 Scaling Operation section for details about the scaler and configuring the scaler for use.
- Refer to the 13.4.3.3 Color Space Conversion (CSC) section for details about CSC and configuring the block for use.
- Refer to the 13.4.3.4 Image De-interlacing section for details about different de-interlacing methods and the configuration for enabling each de-interlacing method for the channels.
- Refer to the 13.4.3.5 High-Res Image Handling section for enabling controls to allow higher than 2K image resolution to be handled by the channel.
- Refer to the 13.4.3.6 Output Buffer Management section for details on how to configure controls for image cropping, flipping, alpha insertion, handle buffer overflows and request higher AXI channel priority for emptying the buffers.

- Refer to the 13.4.3.7 Sourcing Image from Memory section for details on how to configure channel 0 to read and process an image that is sourced from memory.
- Refer to the 13.4.3.8 Flow Control section for details on how to configure the rate of pixel output from this module on to the AXI bus.

Once the above steps are completed, the channel can be enabled and monitored using the following steps:

- Enable the necessary interrupt by writing '1' to the appropriate interrupt enable bits in the `CHNL_IER` register.
- Enable the channel by writing '1' to the `CHNL_CTRL[CHNL_EN]` bit. Then this channel starts accepting images from the selected source.
- Monitor the status of the channel through the `CHNL_STS` register. Each bit provides a status or error indication for the software to take appropriate action. Interrupt status bits can be cleared by writing 1 to respective bits.
- When required, the complete channel can be reset to its default reset state by writing '1' to the `CHNL_CTRL[SW_RST]` bit. Software must keep this bit '1' for at least 5 to 6 NOP cycles to allow reset to propagate and then set this bit to '0' to complete the reset cycle. After the software reset, the channel returns to the same default state as it would have been after a hardware reset. No configuration register is reset during the software reset. Software may additionally clear the `CHNL_CTRL[CHNL_EN]` bit while software reset is asserted to control the restarting of the channel after the software reset completes.

When configuring multiple channels, the above-suggested steps must be carried out for all channels.

4.2.3 ISI width limitation

The line buffer and output buffer storage in one ISI channel is only enough for the 2K (width) image resolution. To be able to process and store a 4K image, line buffer and output buffers from the adjacent channel must be chained with the current channel's line buffer and output buffers respectively. Chaining of line buffers and output buffers has a limitation: only the subsequent channels can be chained. For example, if channel 1 has to handle a 4K image, its line buffer and output buffers can be chained to the line buffer and output buffers from channel 2, respectively, and no other channel. In generic terms, the line buffer and output buffers of channel 'n' can be chained with the line buffer and output buffers, respectively, of channel 'n+1' only.

Note: *This 2K width limitation is valid only for processed camera stream, which means that ISI CSC and/or downscaling feature is enabled for image data processing. For an unprocessed camera stream, the supported width/height can be up to the setting range of the WIDTH/HEIGHT field in the `CHNL_IMG_CFG` register. If a channel's line buffer and output buffers are chained with another channel, that channel should be kept disabled and cannot be used for any image processing.*

Table 31 describes the theoretical maximum image width that ISI can support on i.MX 8MN and i.MX 8MP. But in a real use case, the user must also consider the MIPI Rx subsystem capability and total system bandwidth to see whether such a high resolution can be supported at a certain frame rate.

Table 31. ISI max supported width on i.MX 8MN and i.MX 8MP

SoC	ISI Channel	CHNL_ CTRL[CHAIN_BUF]	Max supported width for processed stream	Max supported width for unprocessed stream
i.MX 8MN	0	/	2048	0x1FFF
i.MX 8MP	0	0	2048	0x1FFF
		1	4096	0x1FFF
	1	/	2048	0x1FFF

4.2.4 Register definition

This section lists some register fields that the user must pay attention to for porting a new camera. For more registers, refer to the i.MX 8MN or i.MX 8MP Reference Manual.

The base address for ISI is:

1. ISI channel 0 for i.MX 8MN CSI-2 Port 1: 32E2_0000h
2. ISI channel 0 for i.MX 8MP CSI-2 Port 1: 32E0_0000h
3. ISI channel 1 for i.MX 8MP CSI-2 Port 2: 32E0_2000h

- CHNL_CTRL register: 0h offset

Table 32. **CHNL_CTRL register**

Field	Description
29 CHNL_BYPASS	Channel bypass enable For certain applications, the ISI module might not be required to do any scaling or color space conversion on the image. For such cases, this bit must be set to bypass the channel pipeline and store the image in memory. The output image format CHNLn_IMG_CTRL[FORMAT] field must be set to the same as the incoming image format. Image flipping, cropping, and alpha insertion can still be carried out. NOTE: For de-interlacing purposes (CHNL_IMG_CTRL[DEINT] set to values other than 00b), the channel is automatically bypassed. NOTE: If this bit is set, CSC is bypassed even if CHNL_IMG_CTRL[CSC_BYP] is not set. 0b - Channel is not bypassed 1b - Channel is bypassed
26-25 CHAIN_BUF	Chain line buffer control For images having higher than 2048 horizontal resolution (pixels), the line buffer RAM (for channel 'n') requires to be concatenated with channel 'n+1' line buffer RAM to provide larger storage. This field controls how the adjacent channel line buffer RAM is chained together. For example, when this field is set to '01', line buffers (for channel 'n' and 'n+1') are chained; the channel 'n+1' cannot be enabled and is blocked for use. 00b - No line buffers chained (supports 2048 or less horizontal resolution for processed stream) 01b - 2 line buffers chained (supports 4096 horizontal resolution for processed stream). Line buffers of channels 'n' and 'n+1' are chained. 10b - Reserved for future use 11b - Reserved for future use
4 SRC_TYPE	Type of selected input image source Defines the type of image input source. Should be set to '1' only when the source of the image is from memory. For all other sources, this bit is '0'. 0b - Image input source is Pixel Link 1b - Image input source is Memory
1-0 SRC	Input image source port selection The source selects the desired input port of the Pixel Link Crossbar from which the image is sourced. This field must not be changed when the channel pipeline is working and must be configured prior to enabling the channel processing. For i.MX 8MN, Setting a value of 1 in channels other than channel 0 will have no effect. Only channel 0, can use pixel link #1 (source from memory). 00b - The image is sourced from input port 0 of the Pixel Link Crossbar. 01b - The image is sourced from input port 1 of the Pixel Link Crossbar (Input port 1 connected to AXI read) 10, 11b - Reserved For i.MX 8MP, Setting a value of 2 in channels other than channel 0 has no effect. Only channel 0, can use pixel link #2 (source from memory). 00b - The image is sourced from input port 0 of the Pixel Link Crossbar 01b - The image is sourced from input port 1 of the Pixel Link Crossbar

Table 32. CHNL_CTRL register...*continued*

Field	Description
	10b - The image is sourced from input port 2 of the Pixel Link Crossbar (Input port 2 connected to AXI read) 11b - Reserved

- CHNL_IMG_CTRL register: 4h offset

Table 33. CHNL_IMG_CTRL register

Field	Description
29-24 FORMAT	Output image format This field specifies the image format to be used when storing the image output from the channel pipeline to memory. The UV byte lanes are synonymous with CbCr byte lanes for YUV output pixel formats. For example, the YUV420_2P* formats should be selected when the output is YCbCr 2-plane 420 output format. 000000b - RGBA8888 - RGB format with alpha in LSB; 8-bits per component. 'A' indicates alpha value. 000001b - ABGR8888 - BGR format with alpha in MSB; 8-bits per component. 'A' indicates alpha value. 000010b - ARGB8888 - RGB format with alpha in MSB; 8-bits per component. 'A' indicates alpha value. 000011b - RGBX888 - RGB format with 8-bits per color component (unpacked and MSB-aligned in 32-bit DWORD). 'X' indicates the waste bits. 000100b - XBGR888 - BGR format with 8-bits per color component (unpacked and LSB aligned in 32-bit DWORD). 'X' indicates the waste bits. 000101b - XRGB888 - RGB format with 8-bits per color component (unpacked and LSB aligned in 32-bit DWORD). 'X' indicates the waste bits. 000110b - RGB888P - RGB format with 8-bits per color component (packed into 24-bits). No waste bits. 000111b - BGR888P - BGR format with 8-bits per color component (packed into 24-bits). No waste bits. 001000b - A2BGR10 - BGR format with 2-bits alpha in MSB; 10-bits per color component. 'A' indicates alpha value.
	001001b - A2RGB10 - RGB format with 2-bits alpha in MSB; 10-bits per color component. 'A' indicates alpha value. 001010b - RGB565 - RGB format with 5-bits of R, B; 6-bits of G (packed into 16-bits WORD). No waste bits. 001011b - RAW8 - 8-bit RAW data packed into 32-bit DWORD 001100b - RAW10 - 10-bit RAW data packed into 16-bit WORD with 6 LSBs waste bits 001101b - RAW10P - 10-bit RAW data packed into 32-bit DWORD 001110b - RAW12 - 12-bit RAW data packed into 16-bit DWORD with 4 LSBs waste bits 001111b - RAW16 - 16-bit RAW data packed into 32-bit DWORD 010000b - YUV444_1P8P with 8-bits per color component; 1-plane, YUV interleaved packed bytes 010001b - YUV444_2P8P with 8-bits per color component; 2-plane, UV interleaved packed bytes 010010b - YUV444_3P8P with 8-bits per color component; 3-plane, non-interleaved packed bytes 010011b - YUV444_1P8 with 8-bits per color component; 1-plane YUV interleaved unpacked bytes (8 MSBs waste bits in 32-bit DWORD) 010100b - YUV444_1P10 with 10-bits per color component; 1-plane, YUV interleaved unpacked bytes (6 LSBs waste bits in 16-bit WORD)

Table 33. CHNL_IMG_CTRL register*...continued*

Field	Description
	010101b - YUV444_2P10 with 10-bits per color component; 2-plane, UV interleaved unpacked bytes (6 LSBs waste bits in 16-bit WORD) 010110b - YUV444_3P10 with 10-bits per color component; 3-plane, non-interleaved unpacked bytes (6 LSBs waste bits in 16-bit WORD)
	010111b - Reserved for future use 011000b - YUV444_1P10P with 10-bits per color component; 1-plane, YUV interleaved packed bytes (2 MSBs waste bits in 32-bit DWORD) 011001b - YUV444_2P10P with 10-bits per color component; 2-plane, UV interleaved packed bytes (2 MSBs waste bits in 32-bit DWORD) 011010b - YUV444_3P10P with 10-bits per color component; 3-plane, non-interleaved packed bytes (2 MSBs waste bits in 32-bit DWORD) 011011b - Reserved for future use 011100b - YUV444_1P12 with 12-bits per color component; 1-plane, YUV interleaved unpacked bytes (4 LSBs waste bits in 16-bit WORD) 011101b - YUV444_2P12 with 12-bits per color component; 2-plane, UV interleaved unpacked bytes (4 LSBs waste bits in 16-bit WORD) 011110b - YUV444_3P12 with 12-bits per color component; 3-plane, non-interleaved unpacked bytes (4 LSBs waste bits in 16-bit WORD) 011111b - Reserved for future use 100000b - YUV422_1P8P with 8-bits per color component; 1-plane, YUV interleaved packed bytes 100001b - YUV422_2P8P with 8-bits per color component; 2-plane, UV interleaved packed bytes 100010b - YUV422_3P8P with 8-bits per color component; 3-plane, non-interleaved packed bytes 100011b - Reserved for future use 100100b - YUV422_1P10 with 10-bits per color component; 1-plane, YUV interleaved unpacked bytes (6 LSBs waste bits in 16-bit WORD)
	100101b - YUV422_2P10 with 10-bits per color component; 2-plane, UV interleaved unpacked bytes (6 LSBs waste bits in 16-bit WORD) 100110b - YUV422_3P10 with 10-bits per color component; 3-plane, non-interleaved unpacked bytes (6 LSBs waste bits in 16-bit WORD) 100111b - Reserved for future use 101000b - YUV422_1P10P with 10-bits per color component; 1-plane, YUV interleaved packed bytes (2 MSBs waste bits in 32-bit DWORD) 101001b - YUV422_2P10P with 10-bits per color component; 2-plane, UV interleaved packed bytes (2 MSBs waste bits in 32-bit DWORD) 101010b - YUV422_3P10P with 10-bits per color component; 3-plane, non-interleaved packed bytes (2 MSBs waste bits in 32-bit DWORD) 101011b - Reserved for future use 101100b - YUV422_1P12 with 12-bits per color component; 1-plane, YUV interleaved unpacked bytes (4 LSBs waste bits in 16-bit WORD) 101101b - YUV422_2P12 with 12-bits per color component; 2-plane, UV interleaved unpacked bytes (4 LSBs waste bits in 16-bit WORD) 101110b - YUV422_3P12 with 12-bits per color component; 3-plane, non-interleaved unpacked bytes (4 LSBs waste bits in 16-bit WORD) 101111b - Reserved for future use 110000b - Reserved for future use 110001b - YUV420_2P8P with 8-bits per color component; 2-plane, UV interleaved packed bytes 110010b - YUV420_3P8P with 8-bits per color component; 3-plane, non-interleaved packed bytes

Table 33. `CHNL_IMG_CTRL` register*...continued*

Field	Description
	110011b - Reserved for future use 110100b - Reserved for future use 110101b - YUV420_2P10 with 10-bits per color component; 2-plane, UV interleaved unpacked bytes (6 LSBs waste bits in 16-bit WORD) 110110b - YUV420_3P10 with 10-bits per color component; 3-plane, non-interleaved unpacked bytes (6 LSBs waste bits in 16-bit WORD) 110111b - Reserved for future use 111000b - Reserved for future use 111001b - YUV420_2P10P with 10-bits per color component; 2-plane, UV interleaved packed bytes (2 MSBs waste bits in 32-bit DWORD) 111010b - YUV420_3P10P with 10-bits per color component; 3-plane, non-interleaved packed bytes (2 MSBs waste bits in 32-bit DWORD) 111011b - Reserved for future use 111100b - Reserved for future use 111101b - YUV420_2P12 with 12-bits per color component; 2-plane, UV interleaved unpacked bytes (4 LSBs waste bits in 16-bit WORD) 111110b - YUV420_3P12 with 12-bits per color component; 3-plane, non-interleaved unpacked bytes (4 LSBs waste bits in 16-bit WORD) 111111b - Reserved for future use
14-12 `DEINT`	De-interlace control Frames that are interlaced when sent over the pixel link require de-interlacing. This field defines the basic de-interlacing method that will be used to combine the interlaced frames. Refer to the ISI Functional Description section in i.MX 8MN or i.MX 8MP Reference Manual for more details on deinterlacing techniques and how to configure the channels to perform de-interlacing. For i.MX 8MN, 000, 001b - No de-interlacing done 010b - Weave de-interlacing (Odd, Even) method used 011b - Weave de-interlacing (Even, Odd) method used 100, 101, 110, 111b - Reserved For i.MX 8MP, 000, 001b - No de-interlacing done 010b - Weave de-interlacing (Odd, Even) method used 011b - Weave de-interlacing (Even, Odd) method used 100b - Blending or linear interpolation (Odd + Even) de-interlacing method used 101b - Blending or linear interpolation (Even + Odd) de-interlacing method used 110, 111b - Line doubling de-interlacing method used. Both Odd and Even fields are doubled.

- `CHNL_IMG_CFG` register: Ch offset

Table 34. `CHNL_IMG_CFG` register

Field	Description
28-16 `HEIGHT`	Input image height (lines) This field provides the height of the input image in terms of the number of lines for both cases when the image is input from a pixel link or memory. The default height is 1080 lines.
12-0 `WIDTH`	Input image width (pixels) This field provides the width of the input image in terms of the number of pixels for both cases when the image is input from a pixel link or memory. The default width is 1920 pixels.

- CHNL_IER register: 10h offset

Table 35. **CHNL_IER register**

Field	Description
30 LINE_RCVD_EN	Line received interrupt enable bit Enables assertion of an interrupt from the channel when the corresponding status bit is asserted. 0b - Interrupt is disabled 1b - Interrupt is enabled
29 FRM_RCVD_EN	Frame received interrupt enable bit Enables assertion of an interrupt from the channel when the corresponding status bit is asserted. 0b - Interrupt is disabled 1b - Interrupt is enabled
22 OFLW_V_BUF_EN	V output buffer overflow interrupt enable bit Enables assertion of an interrupt from the channel when the corresponding status bit is asserted. 0b - Interrupt is disabled 1b - Interrupt is enabled
20 OFLW_U_BUF_EN	U output buffer overflow interrupt enable bit Enables assertion of an interrupt from the channel when the corresponding status bit is asserted. 0b - Interrupt is disabled 1b - Interrupt is enabled
18 OFLW_Y_BUF_EN	Y output buffer overflow interrupt enable bit Enables assertion of an interrupt from the channel when the corresponding status bit is asserted. 0b - Interrupt is disabled 1b - Interrupt is enabled
17 EARLY_VSYNC_ERR_EN	VSYNC timing (Early) error interrupt enable bit Enables assertion of an interrupt from the channel when the corresponding status bit is asserted. 0b - Interrupt is disabled 1b - Interrupt is enabled
16 LATE_VSYNC_ERR_EN	VSYNC timing (Late) error interrupt enable bit Enables assertion of an interrupt from the channel when the corresponding status bit is asserted. 0b - Interrupt is disabled 1b - Interrupt is enabled

- CHNL_STS register: 14h offset

Table 36. **CHNL_STS register**

Field	Description
31 MEM_RD_DONE	Memory read complete interrupt flag Indicates that the channel has read the full image from the input memory location (IN_BUF_ADDR) and software should configure the system for the next image to be read from memory. 0b - Image read from memory not complete or not started 1b - Image read from memory completed
30 LINE_STRD	Line received and stored interrupt flag Indicates that a line has been received and stored in memory. This interrupt flag asserts for every line received and stored in memory. 0b - No new line received 1b - New line received and stored in memory
29	Frame stored successfully interrupt flag

Table 36. CHNL_STS register*...continued*

Field	Description
FRM_STRD	Indicates that a frame has been received and stored in memory. This interrupt flag will assert for every frame received and stored in memory. 0b - No frame being received or in progress 1b - One full frame has been received and stored in memory
22 OFLW_V_BUF	Overflow in U output buffer interrupt flag Indicates overflow due to internal buffers not being read via AXI and are now full and not accepting new valid pixels in the output buffer control stage. The number of bytes lost during the overflow event is also reported in the status register. 0b - No overflow 1b - Overflow has occurred in the channel
20 OFLW_U_BUF	Overflow in U output buffer interrupt flag Indicates overflow due to internal buffers not being read via AXI and are now full and not accepting new valid pixels in the output buffer control stage. The number of bytes lost during the overflow event is also reported in the status register. 0b - No overflow 1b - Overflow has occurred in the channel
18 OFLW_Y_BUF	Overflow in Y/RGB output buffer interrupt flag Indicates overflow due to internal buffers not being read via AXI and are now full and not accepting new valid pixels in the output buffer control stage. The number of bytes lost during the overflow event is also reported in the status register. 0b - No overflow 1b - Overflow has occurred in the channel
17 EARLY_VSYNC_ERR	VSYNC timing (Early) error interrupt flag Indicates that the VSYNC for the frame was detected earlier than expected. The resultant frame will have corrupt data. 0b - No error 1b - VSYNC detected earlier than expected
16 LATE_VSYNC_ERR	VSYNC timing (Late) error interrupt flag Indicates that the VSYNC for the frame was detected later than expected. The resultant frame will have correct data. The extra lines in frame will be ignored. 0b - No error 1b - VSYNC detected later than expected
9 BUF2_ACTIVE	Current frame being stored in Buffer 2 Address Read-only bit that indicates Buffer 2 Address is being used to store the current frame. 0b - Buffer 2 Address inactive 1b - Buffer 2 Address in use
8 BUF1_ACTIVE	Current frame being stored in Buffer 1 Address Read-only bit that indicates Buffer 1 Address is being used to store the current frame. 0b - Buffer 1 Address inactive 1b - Buffer 1 Address in use

5 MIPI capture capabilities

Table 37 shows the MIPI capture capabilities for 8M series processors.

Table 37. Bandwidth of i.MX 8M Series Capture System

SoC	Camera interface	Capture controller	Max total DPHY bandwidth	Max Rx pixel clock
i.MX 8MQ	2 x MIPI CSI-2 (4 lane each)	2 x CSI Bridge	1.5 Gbps x 4 each	333 MHz each
i.MX 8MM	1 x MIPI CSI-2 (4 lane)	1 x CSI Bridge	1.5 Gbps x 4	333 MHz
i.MX 8MN	1 x MIPI CSI-2 (4 lane)	1 x ISI	1.5 Gbps x 4	333 MHz
i.MX 8MP	2 x MIPI CSI-2 (4 lane each)	2 x ISI	1.5 Gbps x 4 each	500 MHz for single camera on CSI Port 1 266 MHz for single camera on CSI Port 2 266 MHz each for dual camera

The incoming camera data rate and pixel clock must be less than or equal to the Rx DPHY bandwidth and pixel clock.

1. camera data rate ≤ Rx DPHY bandwidth
2. camera pixel clock ≤ Rx pixel clock

The camera data rate (bps) and pixel clock (Hz) can be calculated as:

$$\text{Pixel Clock} = \text{H\_total} * \text{V\_total} * \text{FPS}$$

$$\text{Data Rate} = \text{Pixel Clock} * \text{BPP}$$

Where:

- H_total = total line width (in pixels) = active pixels + horizontal blanking pixels
- V_total = total frame height (in lines) = active lines + vertical blanking lines
- FPS = frames per second
- BPP = bits per pixel

Note: *Besides the above bandwidth limitation, the ISI block has an additional width limitation, see Section 4.2.3. ERR011326 and ERR050384 limit the bandwidth of i.MX 8MQ MIPI capture system. Pay attention to the fact that CSI-2 Port 1 has more bandwidth than CSI-2 Port 2 on i.MX 8MQ. All the i.MX 8M series processors can't support multiple camera streams transferred simultaneously via a single MIPI CSI-2 interface. In others words, multi-virtual channels cannot be supported on i.MX 8M series processors.*

6 Porting new camera

This section describes how to port a new camera on i.MX 8M series processors. An AR0144 (1280x800@60fps, RAW12, 2 lanes, 445.5 Mbps per lane) is taken as an example to show how to configure the i.MX 8M series MIPI capture system.

Note: *With the default BSP release, the 8M series MIPI capture system can only support YUV422 format camera input. This section shows how to support other format camera input, such as RAW8, RAW10, RAW12, RGB565, and RGB888.*

6.1 Sensor initialization

The user must make sure that the sensor can work properly through the following steps:

- Check all supply voltages.

- Check sensor MCLK frequency.
- Check reset and power sequence.
- Make sure that I2C communication between the sensor and SoC is functional.
- Make sure that sensor initialization registers are configured correctly.
- Check the sensor data lane and clock lane signals.
- Make sure that the sensor output stream is within the SoC MIPI capture capability (see Section 5).

6.2 MIPI CSI-2 Rx subsystem configuration

Basically, the user must configure the i.MX 8M series MIPI CSI-2 Rx subsystem according to the sensor output. All these configurations can be done in Linux dts and MIPI CSI-2 driver. And the user can check whether these parameters have been configured correctly by reading the values of related register fields (see Section 3.1.6 and Section 3.2.6, “Register Definition”).

- Configure the number of data lanes in CSI-2 DPHY.

Table 38. The number of data lanes in CSI-2 DPHY

SoC	Register	Field	Value
i.MX 8MQ	`CSI2RX_CFG_NUM_LANES`	1-0 csi2rx_cfg_num_lanes	01b
	`CSI2RX_CFG_DISABLE_DATA_LANES`	3-0 csi2rx_cfg_disable_data_lanes	1100b
i.MX 8MM i.MX 8MN i.MX 8MP	`MIPI_CSIx_CSIS_COMMON_CTRL`	9-8 LANE_NUMBER	01b
	`MIPI_CSIx_DPHY_COMMON_CTRL`	4-1 ENABLE_DAT	0011b

Note: *In this section, the “Value” column in all the tables shows the reference setting for AR0144.*

- Configure the high-speed settle timer in CSI-2 DPHY according to the data rate of sensor (see Section 3.1.5 and Section 3.2.5, “DPHY High-Speed Settle Timer”).

Table 39. High-speed settle timer in CSI-2 DPHY according to the data rate of sensor

SoC	Register	Field	Value
i.MX 8MQ	`IOMUXC_GPR_GPR34/41`	7–2 CSI2_x_S_PRG_RXHS_SETTLE	0x6
i.MX 8MM i.MX 8MN i.MX 8MP	`MIPI_CSIx_DPHY_COMMON_CTRL`	31-24 HSSETTLE	0x9

- Configure the image data type, vertical/horizontal image resolution in CSI-2 controller (only for i.MX 8MM, i.MX 8MN and i.MX 8MP).

Table 40. The image data type, vertical/horizontal image resolution in CSI-2 controller

SoC	Register	Field	Value
i.MX 8MM i.MX 8MN i.MX 8MP	`MIPI_CSIx_ISP_CONFIG0`	7–2 DATAFORMAT	0x2C
	`MIPI_CSIx_ISP_RESOLUTION0`	31-16 VRESOL	0x320

Table 40. The image data type, vertical/horizontal image resolution in CSI-2 controller *...continued*

SoC	Register	Field	Value
		15-0 HRESOL	0x500

- Configure the image data type, vertical/horizontal image resolution in Gasket (only for i.MX 8MN and i.MX 8MP).

Table 41. The image data type, vertical/horizontal image resolution in Gasket

SoC	Register	Field	Value
i.MX 8MN i.MX 8MP	`GASKET_x_CTRL`	13-8 GASKET_x_DATA_TYPE	0x2C
	`GASKET_x_HSIZE`	31-0 GASKET_x_HSIZE	0x500
	`GASKET_x_VSIZE`	31-0 GASKET_x_VSIZE	0x320

6.3 Imaging subsystem configuration

The user must also configure the i.MX 8M series Imaging subsystem according to the sensor output, especially format, and image size-related register fields.

6.3.1 i.MX 8MQ/i.MX 8MM CSI bridge configuration

To support RAW, RGB, or YUV format camera input, the following CSI Bridge register fields must be configured correctly.

- `CSI_CR3[SENSOR_16BITS]`
- `CSI_CR18[PARALLEL24_EN]`
- `CSI_CR18[MIPI_DATA_FORMAT]`

Table 42 shows how to configure CSI Bridge to support RAW, RGB, and YUV format sensor.

Table 42. CSI Bridge configuration for RAW, RGB, and YUV format sensor

Sensor Format	CSI_CR3 [SENSOR_16BITS]	CSI_CR18 [PARALLEL24_EN]	CSI_CR18 [MIPI_DATA_FORMAT]
RAW8	0	0	0x2A
RAW10	1	0	0x2B
RAW12	1	0	0x2C
RGB565	1	0	0x22
RGB888	1	1	0x24
YUV422	1	0	0x1E

Note: *There is some difference between i.MX 8MQ and i.MX 8MM regarding the YUV422 format support. On i.MX 8MQ,* `CSI_CR3[SENSOR_16BITS]` *must be set to 1 for YUV422 format sensor. On i.MX 8MM,* `CSI_CR3[SENSOR_16BITS]` *can be set to 0 (single pixel mode) or 1 (dual pixel mode) for YUV422 format sensor. To support dual pixel mode for YUV422 format,* `CSI_CR18[MIPI_DOUBLE_CMPNT]` *should be set to 1 and* `MIPI_CSIx_ISP_CONFIG0[PIXEL_MODE]` *should be set to 01b.*

Moreover, `CSI_IMAG_PARA[IMAGE_WIDTH, IMAGE_HEIGHT]` fields must be configured correctly according to the active sensor output size.

6.3.2 i.MX 8MN/i.MX 8MP ISI configuration

For RGB or YUV format input, ISI block can support format conversion (for example, YUV422 to YUV420) and color space conversion (for example, RGB to YUV). But for RAW format input, it is copied into memory without any pixel changes.

ISI output format is configured in `CHNL_IMG_CTRL[FORMAT]`, and Table 43 shows how to configure ISI to support RAW, RGB, and YUV format sensor.

Table 43. ISI configuration for RAW, RGB, and YUV format sensor

	Sensor Format	CHNL_IMG_CTRL[FORMAT]
No processing	RAW10	001100b - RAW10
	RGB888	000110b - RGB888P
	YUV422	100000b - YUV422_1P8P
Format Conversion	YUV422	110001b - YUV420_2P8P
Color Space Conversion	YUV422	000110b - RGB888P
	RGB565	100000b - YUV422_1P8P

Note: *For no processing case, `CHNL_IMG_CTRL[FORMAT]` must be configured as the same format of the inputting camera sensor. For format conversion and color space conversion cases, Table 43 lists the reference ISI setting. More output formats can be supported, see `CHNL_IMG_CTRL[FORMAT]` fields description in Section 4.2, “Register Definition”.*

Moreover, `CHNL_IMG_CFG[HEIGHT, WIDTH]` fields must be configured correctly according to the active sensor output size.

6.4 Debug tips

When porting a new camera on 8M series processors, the user may encounter the capturing hang issue, which means that no sensor data can be stored in memory. The user can follow the steps below to debug this issue.

- Check MIPI CSI-2 Rx subsystem interrupt status.
  - For i.MX 8MQ, check CSI2RX_IRQ_STATUS register.
    - If DPHY ErrSotHS and/or ErrSotSync_HS error triggers, check whether the `hs_settle` has been configured correctly.
    - If other errors trigger, check whether the camera stream is stable and compliant with the MIPI specification.
  - For i.MX 8MM/i.MX 8MN/i.MX 8MP, check `MIPI_CSIx_DPHY_STATUS` and `MIPI_CSIx_INTERRUPT_SOURCE_0` registers.
    - If a 4-lane camera is connected, `MIPI_CSIx_DPHY_STATUS[STOPSTATEDAT]` must alter between 0x0 and 0xF.
    - If the connected camera works in non-continuous clock mode, `MIPI_CSIx_DPHY_STATUS [STOPSTATECLK]` must alter between 0 and 1; If the connected camera works in continuous clock mode, it should always be 0.
    - If the data lanes or clock lane remain in the stop state or the ULPS state, it normally shows that the camera is not working well. Check the sensor data lane and clock lane signals, and make sure it is compliant with the MIPI specification.

Note: *i.MX 8MM, i.MX 8MN, and i.MX 8MP require the connected camera to work in the LP state before enabling the Rx DPHY. But if the camera works in continuous clock mode, the clock lane may always be in the HS mode. In this case, Rx DPHY may not detect the HS mode and wrongly remains in the stop or ULPS state.*

- If MIPI_CSIx_INTERRUPT_SOURCE_0[ERR_SOT_HS] asserts, check whether the `hs_settle` has been configured correctly.
- If MIPI_CSIx_INTERRUPT_SOURCE_0[ERR_OVER] asserts, tune down the camera data rate.
- If MIPI_CSIx_INTERRUPT_SOURCE_0[FRAME_START/FRAME_END] never asserts or other errors assert, check whether the sensor is working stably.

- Check the Imaging subsystem interrupt status.
  - For i.MX 8MQ and i.MX 8MM, check the `CSI_SR` register.
    - If the capture pipeline is working well, the `DMA_TSF_DONE_FB1` and `DMA_TSF_DONE_FB2` interrupts must trigger in turn, and the time interval between each of these interrupts must be the same as the frame interval of the input camera stream.
    - If `SOF_INT` and/or `EOF_INT` never asserts, it means that CSI Bridge cannot receive any data from the MIPI CSI-2 Rx subsystem. Check whether the camera and MIPI CSI-2 Rx subsystem are configured correctly.
    - If `BASEADDR_CHHANGE_ERROR` triggers many times, it normally shows that the size and/or format-related registers in CSI Bridge are not configured properly. The user can disable this interrupt by writing 0 to `CSI_CR18[BASEADDR_SWITCH_EN]`.
    - If `RF_OR_INT` asserts, try to tune down the camera data rate.

Note: *For i.MX 8MQ, there is a known issue that CSI Bridge Rx FIFO overflow may lead to system hang. Check ERR050384 for more details and the workaround.*

- For i.MX 8MN and i.MX 8MP, check CHNL_STS register.
  - If the capture pipeline is working well, `BUF1_ACTIVE` and `BUF2_ACTIVE` bits must be asserted in turn. And the time interval for each `FRM_STRD` interrupt must be the same as the frame interval of the input camera stream.
  - If `LINE_STRD` and `FRM_STRD` never assert, it means that ISI cannot receive any data from the MIPI CSI-2 Rx subsystem. Check whether the camera and MIPI CSI-2 Rx subsystem are configured correctly.
  - If the `OFLW_Y/U/V_BUF` error asserts, try to tune down the camera data rate.

7 Embedded data support

Some camera sensors may send out embedded data along with image data. As Figure 6 shows, the embedded data is included at the start or end of the frame.

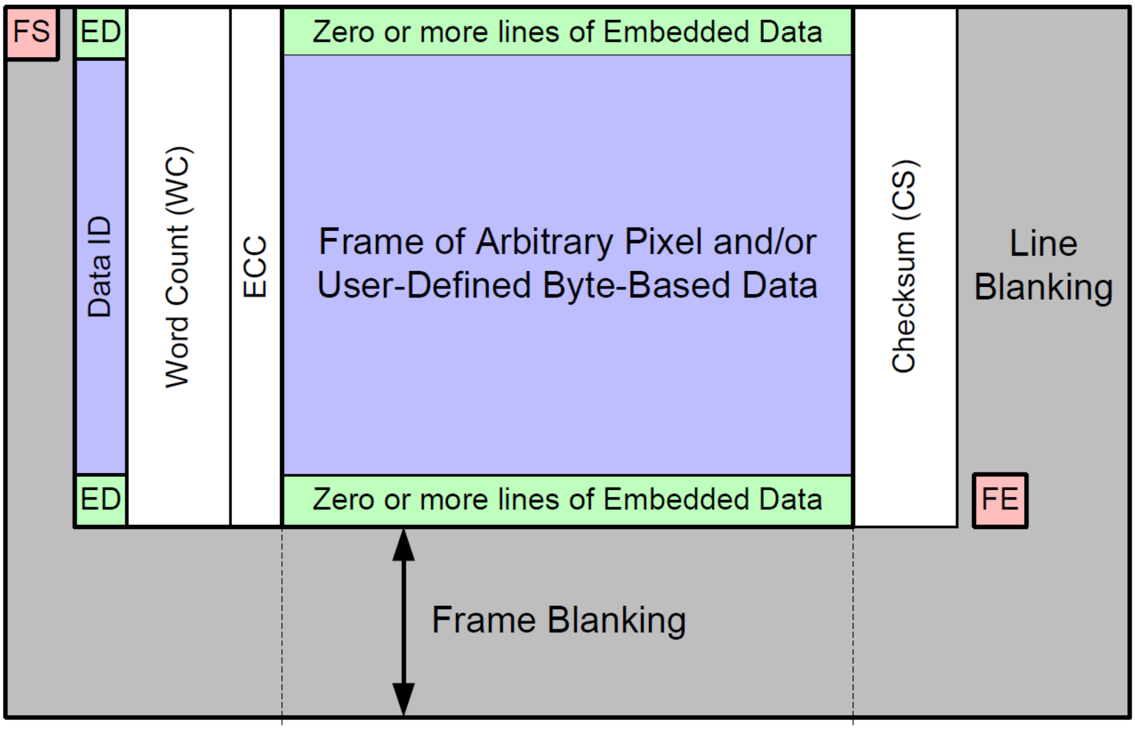
FS
ED
Data ID
Word Count (WC)
ECC
Zero or more lines of Embedded Data
Frame of Arbitrary Pixel and/or User-Defined Byte-Based Data
Zero or more lines of Embedded Data
ED
Checksum (CS)
Line Blanking
FE
Frame Blanking

Figure 6. Structure with Embedded Data at the Beginning and End of the Frame

According to the MIPI CSI-2 specification, the data type of embedded data should be 0x12, which is different from the image data type (from 0x18 to 0x3F). In this case, the camera stream contains multiple types of data, which is defined as data type interleaving mode. However, the 8M series capture system cannot support data type interleaving mode.

Example use case:

Sensor output image frame (640x480, RAW12) and 2 lines embedded data at the start of the frame.

To support the above use case, as a workaround, the user must:

1. Configure the sensor to output embedded data with the image data type (RAW12: 0x2C).
2. Configure the sensor to append zeros for each embedded data to make it aligned with the image pixel (12 bits aligned, {embedded data, 4b’0}).
3. Then the total output of this sensor can be regarded as 640x482 size of RAW12 data. The user can refer to Section 6.2 and Section 6.3 for more information on how to configure the Rx capture system to support this case.

Note: *If the sensor can only send out embedded data with 0x12 data type, then it can’t be supported on i.MX 8M series processors.*

8 Interlaced sensor support and De-interlacing

All the i.MX 8M series processors can support interlaced MIPI cameras. But only on i.MX 8MN and i.MX 8MP, hardware de-interlacing feature can be supported.

Example use case:

Sensor output interlaced odd/even frames (720x288, yuv422).

User can refer to the following steps for how to enable ISI weaving de-interlacing for this use case.

1. Refer to Section 6.2 and Section 6.3 for how to receive 720x288, yuv422 frames.
2. Configure `GASKET_x_CTRL[GASKET_x_INTER_MODE]` field to 01b.
3. Configure `CHNL_IMG_CTRL[DEINT]` field to 10b or 11b, then the de-interlaced frame is stored into memory for every two `FRM_STRD` interrupts.

9 Revision history

Table 44. Revision history

Revision number	Date	Substantive changes
0	15 March 2023	Initial release

10 Legal information

10.1 Definitions

Draft — A draft status on a document indicates that the content is still under internal review and subject to formal approval, which may result in modifications or additions. NXP Semiconductors does not give any representations or warranties as to the accuracy or completeness of information included in a draft version of a document and shall have no liability for the consequences of use of such information.

10.2 Disclaimers

Limited warranty and liability — Information in this document is believed to be accurate and reliable. However, NXP Semiconductors does not give any representations or warranties, expressed or implied, as to the accuracy or completeness of such information and shall have no liability for the consequences of use of such information. NXP Semiconductors takes no responsibility for the content in this document if provided by an information source outside of NXP Semiconductors.

In no event shall NXP Semiconductors be liable for any indirect, incidental, punitive, special or consequential damages (including - without limitation - lost profits, lost savings, business interruption, costs related to the removal or replacement of any products or rework charges) whether or not such damages are based on tort (including negligence), warranty, breach of contract or any other legal theory.

Notwithstanding any damages that customer might incur for any reason whatsoever, NXP Semiconductors' aggregate and cumulative liability towards customer for the products described herein shall be limited in accordance with the Terms and conditions of commercial sale of NXP Semiconductors.

Right to make changes — NXP Semiconductors reserves the right to make changes to information published in this document, including without limitation specifications and product descriptions, at any time and without notice. This document supersedes and replaces all information supplied prior to the publication hereof.

Suitability for use — NXP Semiconductors products are not designed, authorized or warranted to be suitable for use in life support, life-critical or safety-critical systems or equipment, nor in applications where failure or malfunction of an NXP Semiconductors product can reasonably be expected to result in personal injury, death or severe property or environmental damage. NXP Semiconductors and its suppliers accept no liability for inclusion and/or use of NXP Semiconductors products in such equipment or applications and therefore such inclusion and/or use is at the customer's own risk.

Applications — Applications that are described herein for any of these products are for illustrative purposes only. NXP Semiconductors makes no representation or warranty that such applications will be suitable for the specified use without further testing or modification.

Customers are responsible for the design and operation of their applications and products using NXP Semiconductors products, and NXP Semiconductors accepts no liability for any assistance with applications or customer product design. It is customer's sole responsibility to determine whether the NXP Semiconductors product is suitable and fit for the customer's applications and products planned, as well as for the planned application and use of customer's third party customer(s). Customers should provide appropriate design and operating safeguards to minimize the risks associated with their applications and products.

NXP Semiconductors does not accept any liability related to any default, damage, costs or problem which is based on any weakness or default in the customer's applications or products, or the application or use by customer's third party customer(s). Customer is responsible for doing all necessary testing for the customer's applications and products using NXP Semiconductors products in order to avoid a default of the applications and the products or of the application or use by customer's third party customer(s). NXP does not accept any liability in this respect.

Terms and conditions of commercial sale — NXP Semiconductors products are sold subject to the general terms and conditions of commercial sale, as published at http://www.nxp.com/profile/terms, unless otherwise agreed in a valid written individual agreement. In case an individual agreement is concluded only the terms and conditions of the respective agreement shall apply. NXP Semiconductors hereby expressly objects to applying the customer's general terms and conditions with regard to the purchase of NXP Semiconductors products by customer.

Export control — This document as well as the item(s) described herein may be subject to export control regulations. Export might require a prior authorization from competent authorities.

Suitability for use in non-automotive qualified products — Unless this data sheet expressly states that this specific NXP Semiconductors product is automotive qualified, the product is not suitable for automotive use. It is neither qualified nor tested in accordance with automotive testing or application requirements. NXP Semiconductors accepts no liability for inclusion and/or use of non-automotive qualified products in automotive equipment or applications.

In the event that customer uses the product for design-in and use in automotive applications to automotive specifications and standards, customer (a) shall use the product without NXP Semiconductors' warranty of the product for such automotive applications, use and specifications, and (b) whenever customer uses the product for automotive applications beyond NXP Semiconductors' specifications such use shall be solely at customer's own risk, and (c) customer fully indemnifies NXP Semiconductors for any liability, damages or failed product claims resulting from customer design and use of the product for automotive applications beyond NXP Semiconductors' standard warranty and NXP Semiconductors' product specifications.

Translations — A non-English (translated) version of a document, including the legal information in that document, is for reference only. The English version shall prevail in case of any discrepancy between the translated and English versions.

Security — Customer understands that all NXP products may be subject to unidentified vulnerabilities or may support established security standards or specifications with known limitations. Customer is responsible for the design and operation of its applications and products throughout their lifecycles to reduce the effect of these vulnerabilities on customer's applications and products. Customer's responsibility also extends to other open and/or proprietary technologies supported by NXP products for use in customer's applications. NXP accepts no liability for any vulnerability. Customer should regularly check security updates from NXP and follow up appropriately.

Customer shall select products with security features that best meet rules, regulations, and standards of the intended application and make the ultimate design decisions regarding its products and is solely responsible for compliance with all legal, regulatory, and security related requirements concerning its products, regardless of any information or support that may be provided by NXP.

NXP has a Product Security Incident Response Team (PSIRT) (reachable at PSIRT@nxp.com) that manages the investigation, reporting, and solution release to security vulnerabilities of NXP products.

10.3 Trademarks

Notice: All referenced brands, product names, service names, and trademarks are the property of their respective owners.

NXP — wordmark and logo are trademarks of NXP B.V.

Contents

Please be aware that important notices concerning this document and the product(s) described herein, have been included in section 'Legal information'.

For more information, please visit: http://www.nxp.com

Date of release: 15 March 2023
Document identifier: AN13857