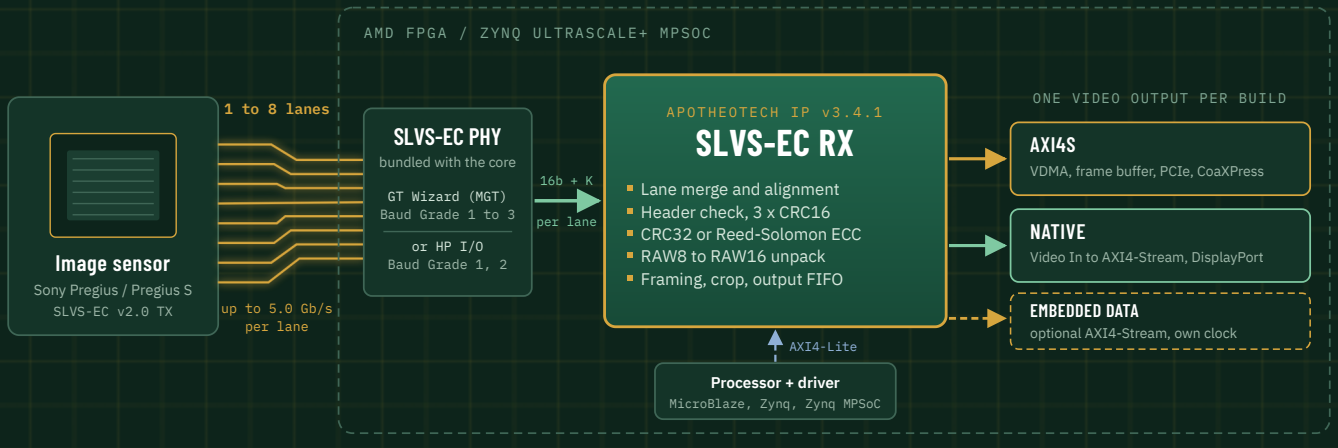
APOTHEOTECH

IP CORE DATASHEET
APT-DS-SLVS-RX-0001
Rev 2.1 · IP v3.4.1
November 2026

SLVS-EC V2.0 RX LINK LAYER FOR AMD FPGAS

SLVS-EC Receiver IP Core

Up to eight lanes of Sony image sensor data at up to 5.0 Gb/s per lane, delivered as AXI4-Stream or native video with header CRC, payload CRC32 or Reed-Solomon correction built in, and the sensor's embedded data lines on a stream of their own. Bundled PHY on FPGA transceivers or HP I/O.



1 to 8
lanes, build-time

40 Gb/s
aggregate line rate

RAW8 to 16
run-time pixel depth

MGT or HP I/O
bundled PHY

CONTENTS

Contents

01 Overview

1.1 Introduction
1.2 Features
1.3 Key specifications
1.4 Throughput
1.5 Supported image sensors
1.6 Applications

02 Functional Description

2.1 Architecture
2.2 PHY options
2.3 Packet handling
2.4 Payload protection
2.5 Pixel formats
2.6 Video outputs
2.7 Embedded data output
2.8 Framing and crop
2.9 FIFO sizing
2.10 Overflow and recovery
2.11 Clocks and reset

03 Interfaces and Timing

3.1 IP customization parameters
3.2 Interface overview
3.3 Port descriptions
3.4 Output timing
3.5 Integration notes

04 Register Map

4.1 Register interface
4.2 Register summary
4.3 Identification
4.4 Link configuration
4.5 Error counters
4.6 Video output
4.7 Embedded data output
4.8 Programming ECC_BLOCK
4.9 Programming sequence

05 Example Designs and Software

5.1 Evaluation platforms
5.2 Demo bitstreams
5.3 Example designs
5.4 Bring-up flow
5.5 Software
5.6 Verification

06 Resources and Ordering

6.1 Supported devices
6.2 Resource utilization
6.3 Deliverables
6.4 Evaluation and ordering
6.5 Revision history
6.6 Notices

© 2026 ApotheoTech LLC | apotheotech.com | info@apotheotech.com

SECTION 01

Overview

A complete SLVS-EC v2.0 receive link layer for AMD FPGAs, from transceiver lanes to frame-ready video.

1.1 Introduction

The ApotheoTech SLVS-EC Receiver IP Core implements the receive link layer of the JIIA EVI-001-2019 SLVS-EC Interface Standard v2.0, the embedded-clock serial interface of Sony Pregius and Pregius S global shutter image sensors. It comes with its own physical layer: the APT-SLVS-EC GT Wizard on AMD GTH or GTY transceivers, or the SLVS-EC HP I/O PHY on high-performance I/O. The core checks every packet header, verifies or corrects the payload, unpacks RAW8 to RAW16 pixels and hands your design whole frames on the video interface you select when you generate the IP.

Two output options cover the usual integration paths. An AXI4-Stream video master on its own clock feeds a VDMA, a frame buffer, a PCIe DMA or a CoaXPress bridge. A native video output connects straight to AMD Video In to AXI4-Stream and DisplayPort pipelines. A compact AXI4-Lite register file identifies the build, configures the link, crops the image and reports error and overflow counts. An optional second AXI4-Stream carries the sensor's embedded data lines, so per-frame metadata reaches software without touching the image path. Every clock-domain crossing is inside the core.

The core ships with its PHY, example designs, a software driver and demo bitstreams, so a sensor can be streaming on a known-good platform before your own hardware is ready.

1.2 Features

LINK LAYER

- Receive link layer of the SLVS-EC Interface Standard v2.0 (JIIA EVI-001-2019)
- 1, 2, 4, 6 or 8 lanes, selected at build time
- Baud Grade 1, 2 and 3: up to 5.0 Gb/s per lane, 40 Gb/s aggregate
- Bundled PHY: APT-SLVS-EC GT Wizard on GTH or GTY up to Baud Grade 3, or SLVS-EC HP I/O up to Baud Grade 2
- Triple packet header, each copy checked by CRC16; the first good copy is used
- Lane merge, byte realignment and Pad Code removal across all lanes
- Frame Start, Frame End and Line Valid decoding into whole frames

PAYLOAD INTEGRITY

- CRC build: payload CRC32 footer check
- ECC build: Reed-Solomon payload correction, ECC Option 1 (1 byte per block) or Option 2 (up to 2 bytes per block)
- Header CRC counters for bad copies and lost headers; payload CRC32 counter with the Line Number of the last failing line
- Corrected-symbol counters, per frame and running
- Uncorrectable blocks passed on unchanged and counted, never hidden

VIDEO OUTPUT

- AXI4-Stream video (tuser = start of frame, tlast = end of line) on an independent clock
- Native video with vsync, hsync, hblank, vblank and gap-free active lines

INTEGRATION

- AXI4-Lite control with ID, version and build discovery
- Lane, register and stream clocks fully independent; no user CDC constraints

© 2026 ApotheoTech LLC | apotheotech.com | info@apotheotech.com

- RAW8, RAW10, RAW12, RAW14 and RAW16, selectable at run time
- 2 x LANE_NUM pixels per beat in MSB-justified 16-bit containers
- Optional embedded data output: the sensor's metadata lines on their own AXI4-Stream, clock and FIFO, with either video output
- Crop by line skip, width and height, updated frame by frame while streaming
- Clean recovery from output overflow without a reset
- Packaged for Vivado IP Integrator with a customization GUI
- Software driver for setup and control on MicroBlaze, Zynq-7000 and Zynq UltraScale+ MPSoC
- Example designs for Zynq UltraScale+ MPSoC, Kria K26 and Kintex UltraScale+

1.3 Key specifications

Table 1-1. Key specifications of the SLVS-EC Receiver IP Core v3.4.1.

Standard	JIIA EVI-001-2019, SLVS-EC Interface Standard v2.0, RX link layer
Lanes	1, 2, 4, 6 or 8 (build-time parameter `LANE_NUM`)
Line rate per lane	Baud Grade 1: 1.152 to 1.2 Gb/s (HP I/O or MGT PHY) Baud Grade 2: 2.304 to 2.5 Gb/s (HP I/O or MGT PHY) Baud Grade 3: 4.608 to 5.0 Gb/s (MGT PHY)
PHY	Bundled with the core: APT-SLVS-EC GT Wizard on GTH or GTY transceivers (Baud Grade 1 to 3), or SLVS-EC HP I/O PHY (Baud Grade 1 and 2)
Aggregate line rate	Up to 40 Gb/s (8 lanes at 5.0 Gb/s); 32 Gb/s of payload after 8b/10b
Line coding	8b/10b, decoded by the bundled PHY
Pixel formats	RAW8, RAW10, RAW12, RAW14, RAW16, selected by register
Line length	Up to 65,532 pixels: a multiple of 4 at 1, 2, 4 and 6 lanes, a multiple of 8 at 8 lanes; by register or from the packet header
Header protection	Three header copies per packet, CRC16 each; bad copies and lost headers counted
Payload protection	CRC build: CRC32 footer check ECC build: Reed-Solomon, 2 or 4 parity bytes per 224-byte block
Video output	AXI4-Stream or native video (build-time parameter `OUT_MODE`)
Embedded data	Optional AXI4-Stream `M_AXIS_EBD` on its own clock, 32 to 16,384-word FIFO (build-time parameter `EBD_OUT`); never in the video output
Output width	32 x LANE_NUM bits: 2 x LANE_NUM pixels per beat, 16-bit MSB-justified containers
Output buffer	128 to 16,384 words (one word per beat), default 2,048
Control interface	AXI4-Lite slave, 32-bit data, 24 registers in a 256-byte window
Clocks	Lane clock from the PHY = line rate / 20 (237.6 MHz at 4.752 Gb/s); register clock and stream clocks independent
Target devices	AMD Zynq UltraScale+ MPSoC, Kria K26 SOM, Kintex UltraScale+, Artix UltraScale+
Software	Driver for setup and control on MicroBlaze, Zynq-7000 and Zynq UltraScale+ MPSoC, supplied on purchase
Design flow	AMD Vivado Design Suite, IP Integrator or RTL instantiation

© 2026 ApotheoTech LLC | apotheotech.com | info@apotheotech.com

1.4 Throughput

The receiver keeps up with the link in every configuration. Each lane delivers 16 decoded bits per lane clock, and the core emits a beat of 2 x LANE_NUM pixels as soon as the bits for it have arrived, so the sensor never has to be throttled for the core.

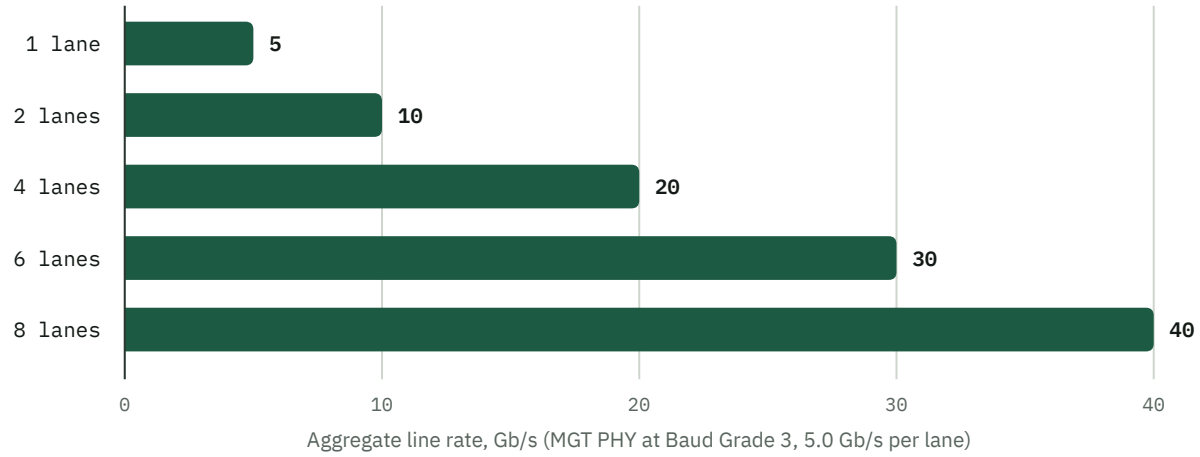


Figure 1-1. Aggregate line rate by lane count at the top of Baud Grade 3 with the MGT PHY. Hover a bar for the payload rate after 8b/10b.

Table 1-2. Maximum aggregate line rate in Gb/s, by Baud Grade and lane count, with the PHY that reaches it. Pixel payload is 80 % of the line rate after 8b/10b decoding.

PER LANE	PHY	1 LANE	2 LANES	4 LANES	6 LANES	8 LANES
Baud Grade 1 1.152 to 1.2 Gb/s	HP I/O or MGT	1.2	2.4	4.8	7.2	9.6
Baud Grade 2 2.304 to 2.5 Gb/s	HP I/O or MGT	2.5	5	10	15	20
Baud Grade 3 4.608 to 5 Gb/s	MGT	5	10	20	30	40

1.5 Supported image sensors

The receiver is validated with the Sony sensors below. It follows the standard rather than a sensor, so other SLVS-EC sensors work with the right register settings; ApotheoTech adds sensor-specific example software and bring-up support through an NRE engagement. Baud Grade 3 modes use the MGT PHY; Baud Grade 1 and 2 modes also run on HP I/O.

Table 1-3. Validated Sony SLVS-EC sensors. Frame rates are the sensor maximum in all-pixel mode at 8 / 10 / 12-bit, from Sony product information.

SENSOR	FAMILY	RESOLUTION	SLVS-EC	MAX FPS (8 / 10 / 12-BIT)
IMX530	Pregius S, 4th gen	24.55 MP, 5328 x 4608	8 lanes, Baud Grade 3	105 / 101 / 73

© 2026 ApotheoTech LLC | apotheotech.com | info@apotheotech.com

SENSOR	FAMILY	RESOLUTION	SLVS-EC	MAX FPS (8 / 10 / 12-BIT)
IMX547	Pregius S, 4th gen	5.1 MP, 2472 x 2064	2 lanes, Baud Grade 3	122.9 / 122.1 / 82.4
IMX421	Pregius, 3rd gen	2.83 MP, 1936 x 1464	8 lanes, Baud Grade 2	409.2 / 371.8 / 231.2
Other SLVS-EC sensors	Sony SLVS-EC v1.x and v2.0 sensors			NRE engagement

1.6 Applications

Systems that need high-bandwidth, low-latency capture from Sony Pregius and Pregius S sensors:

Industrial inspection
Global shutter capture of fast-moving parts

Semiconductor metrology
High-resolution wafer and panel inspection

High-speed capture
Hundreds of frames per second at full resolution

Motion analysis
Distortion-free capture of fast events

Scientific imaging
Raw sensor data with error accounting

Machine vision and robotics
Embedded vision on Zynq UltraScale+ and Kria

Camera bridges
SLVS-EC to CoaXPress, PCIe or Ethernet

Defense and aerospace imaging
Deterministic FPGA pipelines with full visibility

© 2026 ApotheoTech LLC | apotheotech.com | info@apotheotech.com

SECTION 02

Functional Description

How the receiver turns SLVS-EC packets into video: the PHY, the datapath, payload protection, pixel formats, the two video outputs, the embedded data output, crop, buffering and clocks.

2.1 Architecture

The core is the link layer of an SLVS-EC receiver and comes with the physical layer it runs on. The bundled PHY recovers the clock and data on each lane, aligns and decodes the 8b/10b symbols and deskews the lanes; the core takes the decoded lanes from there. Every packet then passes through the stages below and leaves as video on the output selected at build time.

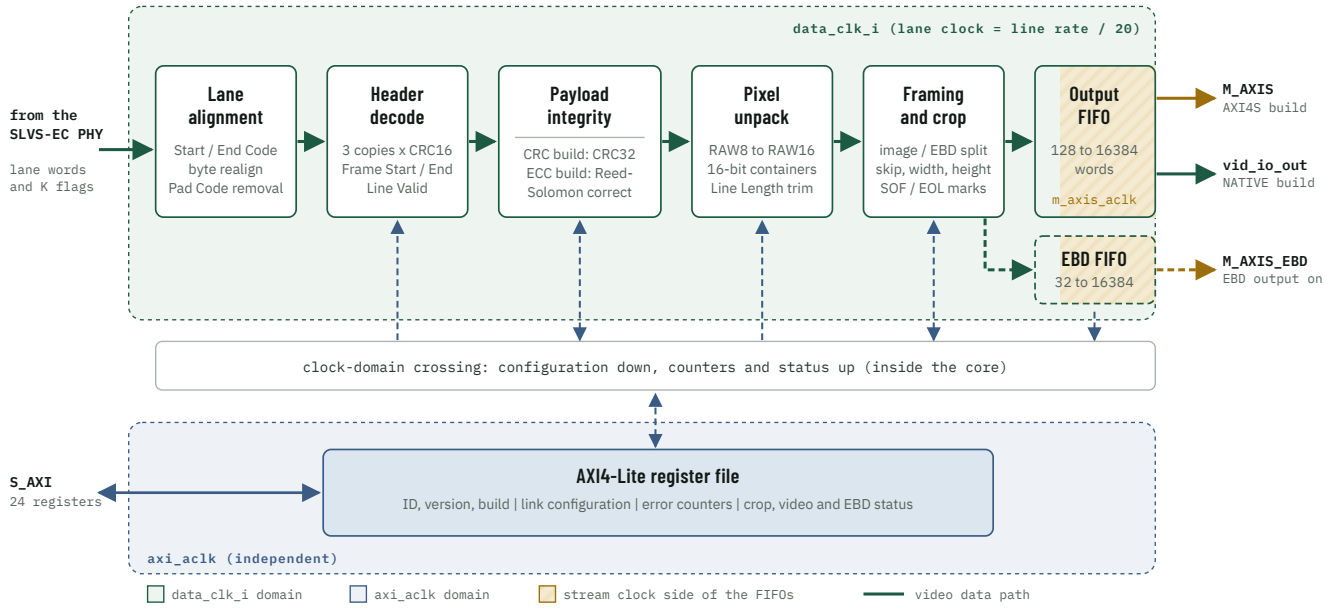


Figure 2-1. Functional block diagram. The video path runs on the lane clock; the register file has its own clock; an AXI4S build reads its output FIFO on the stream clock, and the embedded data output, when built, has its own FIFO and clock. All crossings are inside the core.

01 Lane alignment

Finds the Start Code on the lanes, realigns bytes that arrive split across lane words, and removes Pad Codes, so each packet starts on a clean word boundary at any lane count.

02 Header decode

Checks the three copies of the packet header with CRC16 and uses the first good copy. Frame Start, Frame End and Line Valid come from it.

© 2026 ApotheoTech LLC | apotheotech.com | info@apotheotech.com

03 Payload integrity

The CRC build checks the CRC32 footer of each packet. The ECC build corrects the payload with Reed-Solomon codes and strips the parity bytes.

04 Pixel unpack

Converts the payload bytes back into RAW8 to RAW16 pixels, places each in a 16-bit container and cuts the line at Line Length so payload stuffing never reaches the output.

05 Framing and crop

Builds frames from Frame Start to Frame End, sends image lines to the video output and embedded data lines to their own stream, applies the crop registers and marks start of frame and end of line.

06 Output FIFOs

A line buffer for native video, or a clock-crossing FIFO for AXI4-Stream, plus a second clock-crossing FIFO for the embedded data output when it is built. Both are sized in the IP GUI.

2.2 PHY options

The PHY is delivered with the core and is not configured by the user. Pick the one that fits the sensor mode and the board:

Table 2-1. Bundled PHY options.

PHY	FPGA RESOURCE	BAUD GRADES	MAX PER LANE	USE IT WHEN
APT-SLVS-EC GT Wizard	GTH or GTY transceivers (MGT)	1, 2, 3	5.0 Gb/s	The sensor mode needs Baud Grade 3, or the lanes are routed to transceiver pins
SLVS-EC HP I/O	High-performance (HP) I/O banks, no transceivers	1, 2	2.5 Gb/s	The sensor mode runs at Baud Grade 1 or 2, transceivers are taken, or the lanes are routed to HP I/O

Both PHYs present the same interface to the core, so the rest of the design, the registers and the software do not change with the choice.

2.3 Packet handling

An SLVS-EC sensor sends one packet per line. Each packet carries a header (sent three times), the packed pixel payload padded to a multiple of 16 bytes, and, depending on the sensor mode, a CRC32 footer or Reed-Solomon parity. The receiver uses these header fields:

Table 2-2. Packet header fields used by the receiver.

FIELD	USE IN THE CORE
Frame Start / Frame End	Open and close a frame. Every frame on the output begins with a start-of-frame mark.
Line Valid	Marks packets that carry image lines. Blanking packets are not sent as video.
Embedded data line	Lines of sensor metadata. Never part of the video output; delivered on the embedded data output when it is built.

© 2026 ApotheoTech LLC | apotheotech.com | info@apotheotech.com

FIELD	USE IN THE CORE
Line length (sensor-specific header field)	When the sensor reports the line length in the header, the core uses it instead of the `LINE_LENGTH` register.

One or two bad header copies are recovered from a good one and counted in `HDR_CRC_ERR_CNT[15:0]`, an early sign of a marginal link. A packet whose three header copies all fail CRC16 is lost: it is not used as video and is counted in `HDR_CRC_ERR_CNT[31:16]`. If that packet was a Frame End the frame joins the next one; if it was a Frame Start that frame is dropped. The next good Frame Start resynchronises the output, so a single bad header never shifts the image.

2.4 Payload protection

SLVS-EC sensors protect the payload in one of two ways, chosen by the sensor mode. The receiver comes in a matching build, chosen with the **Payload protection** parameter (`ECC_SEL`). The lane count, ECC Option and CRC Option must match on both ends of the link.

> **RULE**
>
> **Keep the protection of the build on.** The CRC build runs with CRC Option on (`CRC_OPTION` = 1); the ECC build runs with ECC Option 1 or 2 (`ECC_OPTION`). The option of the other build has no effect. The core does not refuse a 0, but the supplied driver checks the setting against the build when an image stream starts and prints a warning if the protection is off or set on the wrong build. With the protection off the image still arrives; what is missing is the check.

Table 2-3. The two payload protection builds.

	CRC BUILD (ECC_SEL = 0)	ECC BUILD (ECC_SEL = 1)
Sensor mode	CRC Option on (CRC32 footer)	ECC Option 1 or 2
What the core does	Checks the CRC32 footer of every packet	Corrects the payload block by block and removes the parity
Correction	Detection only	Option 1: 1 byte per block (2 parity bytes) Option 2: up to 2 bytes per block (4 parity bytes)
Configuration	`CRC_OPTION`	`ECC_OPTION`, `ECC_BLOCK`
Error accounting	`PAY_CRC_ERR_CNT`, and the Line Number of the last failing line in `CRC_ERR_LINE`	`C_ERR_CNT`, `FRAME_ERR_CNT` and `ECC_FAIL_CNT`
Both builds	Bad header copies and lost headers in `HDR_CRC_ERR_CNT`; one sticky flag for software to poll in `CRC_ERR_LINE[31]`	

With ECC, the payload of every line is split into Basic Blocks of 224 information bytes and a last block holding the rest, each followed by its parity. The core needs to know where the last block starts; the `ECC_BLOCK` register gives that from Line Length and Pixel Bit (the driver computes it, see Programming ECC_BLOCK).

© 2026 ApotheoTech LLC | apotheotech.com | info@apotheotech.com

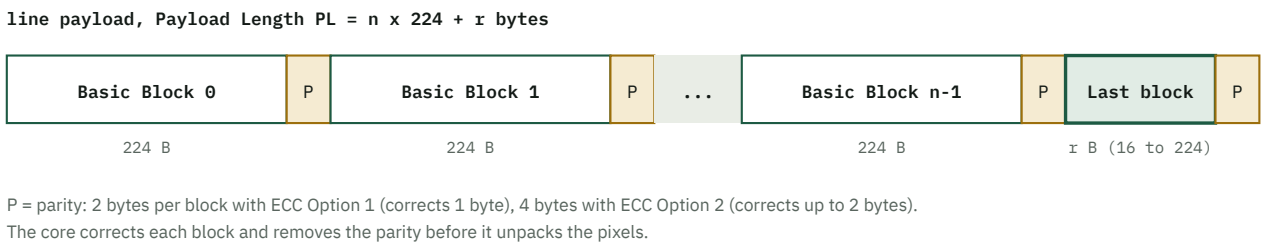


Figure 2-2. Reed-Solomon block structure of one line payload in the ECC build.

A block the code cannot correct is passed on unchanged and counted in `ECC_FAIL_CNT`; the core never alters data it cannot fix. On a healthy link `ECC_FAIL_CNT` stays at 0 and `C_ERR_CNT` grows slowly, if at all. A fast-growing corrected count is an early warning of a signal integrity problem, well before images degrade.

2.5 Pixel formats

The sensor packs pixels into bytes before sending them. Pixels wider than 8 bits send their top 8 bits as one byte each and bundle the remaining low bits of a small group into the bytes that follow. The core reverses this for every lane count and pixel depth.

Table 2-4. Pixel packing on the link and output beat rate during a line.

PIXEL BIT	PACKING ON THE LINK	OUTPUT BEATS DURING A LINE
RAW8	1 pixel per byte	every lane clock
RAW10	4 pixels in 5 bytes	4 of 5 lane clocks
RAW12	2 pixels in 3 bytes	2 of 3 lane clocks
RAW14	4 pixels in 7 bytes	4 of 7 lane clocks
RAW16	1 pixel in 2 bytes, MSB first	1 of 2 lane clocks

Both outputs use the same format. A beat carries 2 x LANE_NUM pixels, each in its own 16-bit container. Container *k* occupies bits 16*k*+15 down to 16*k* and holds the *k*-th pixel of the beat, so container 0 is the earliest pixel. Pixels are MSB-justified: pixel bit Pixel Bit - 1 sits at container bit 15 and the unused low bits are 0. A RAW12 pixel therefore reads as its 12-bit value shifted left by 4, and software can treat every depth as 16-bit data.

© 2026 ApotheoTech LLC | apotheotech.com | info@apotheotech.com

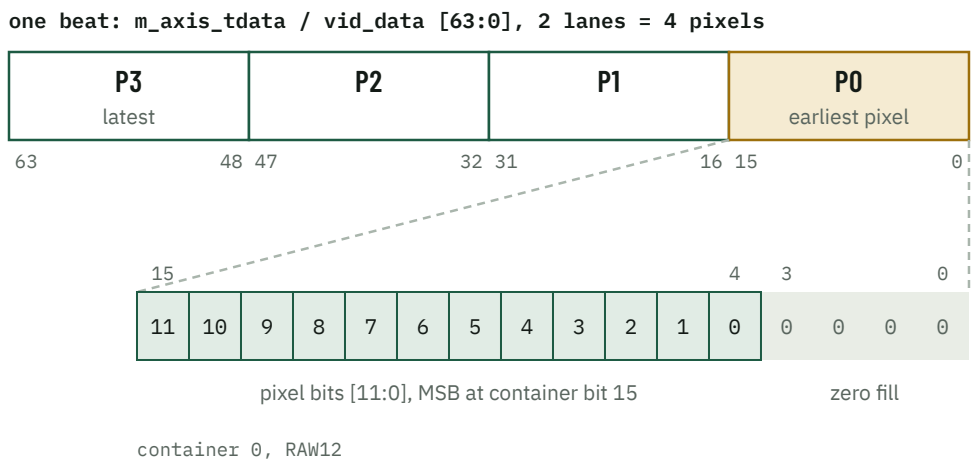


Figure 2-3. Output beat of a 2-lane build and the 16-bit container of a RAW12 pixel.

2.5.1 Line Length

`LINE_LENGTH` is the sensor's pixels per line, for image lines and embedded data lines alike, and has to match the sensor mode:

Table 2-5. Line Length rule by lane count.

LANES	LINE LENGTH	PIXELS PER BEAT
1, 2, 4, 6	Any multiple of 4, up to 65,532	2, 4, 8, 12
8	Any multiple of 8, up to 65,528	16

A line that is not a whole number of beats ends on a partial beat. The pixels of the line are all there; the containers after the last pixel of that beat carry non-image data, so a consumer that needs a clean edge crops to the line width. At 8 lanes, Line Lengths that are a multiple of 4 but not of 8 are available as an NRE item.

2.6 Video outputs

One parameter, **Output** (`OUT_MODE`), selects which video interface is built. Only the ports of the selected output appear on the IP. Both carry the same pixel format and the same frames.

Table 2-6. Choosing an output.

OUTPUT	USE IT WHEN	CLOCK	BUFFER
AXI4S `OUT_MODE 1`	The next block is an AXI4-Stream sink on its own clock: AXI VDMA, a frame buffer writer, an HLS pipeline, a PCIe DMA or a CoaXPress transmitter	`m_axis_aclk`, any frequency	Clock-crossing FIFO, absorbs `tready` stalls
NATIVE `OUT_MODE 0`	The next block takes native video: AMD Video In to AXI4-Stream, or a pipeline on the lane clock	`data_clk_i`	Line buffer, one line of latency

© 2026 ApotheoTech LLC | apotheotech.com | info@apotheotech.com

2.6.1 AXI4S

An AXI4-Stream video master in the AMD video convention. `tuser` marks the first beat of every frame and `tlast` the last beat of every line. Between lines and frames `tvalid` is low; while the FIFO has room, `tready` may stall the stream at any time without losing data.

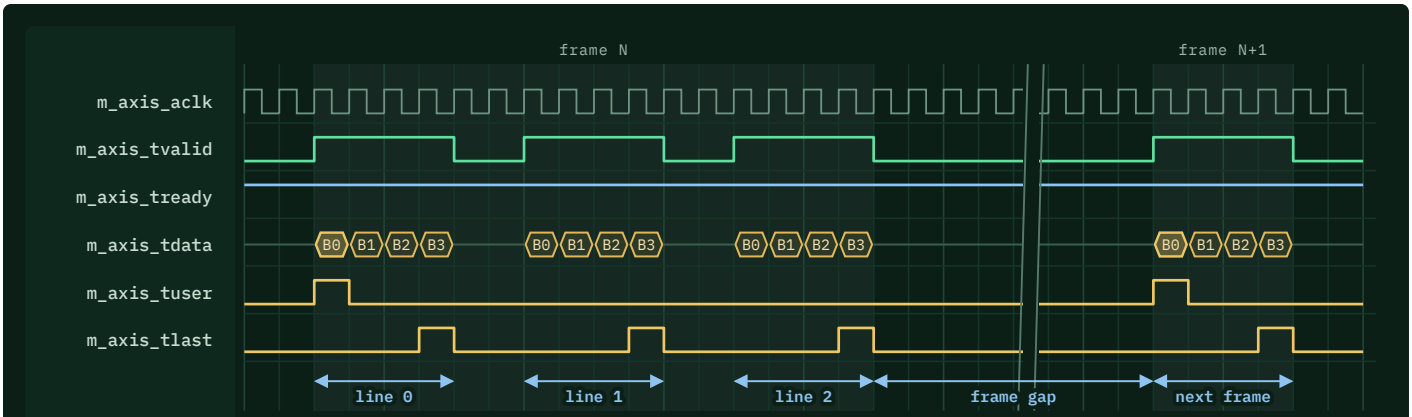


Figure 2-4. AXI4S output over one frame of three short lines (4 beats each), then the first beats of the next frame. Every port of the interface is shown.

2.6.2 NATIVE

Native video with the signal set AMD video IP expects. Each line leaves as one unbroken run of `vid_active_video`; `vid_hsync` pulses after every line and `vid_vsync` after every frame, at the start of the blanking that follows.

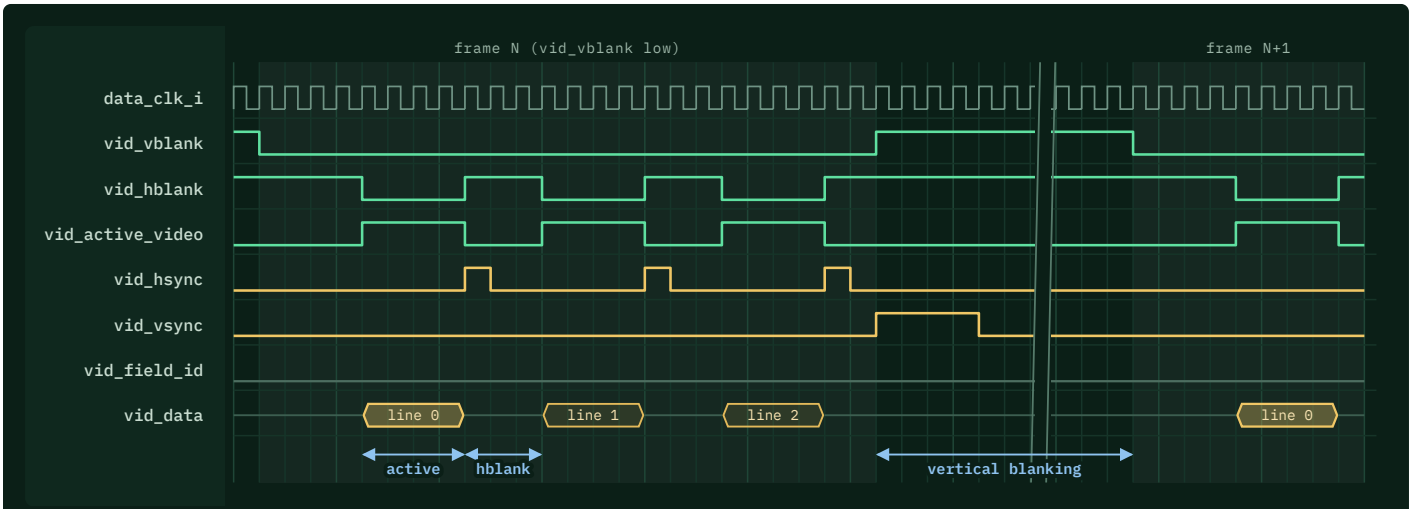


Figure 2-5. NATIVE output over one frame of three short lines, the vertical blanking, and the start of the next frame. Every port of `vid_io_out` is shown.

Clock-by-clock timing for both outputs is in Output timing.

© 2026 ApotheoTech LLC | apotheotech.com | info@apotheotech.com

2.7 Embedded data output

Sony sensors can send lines of metadata with each frame: register settings, frame counters, exposure and similar information. The video outputs carry image lines only. With **Embedded data output** on in the IP GUI (`EBD_OUT` = 1), the embedded data lines leave the core on an AXI4-Stream of their own, `M_AXIS_EBD` , with its own clock and its own FIFO. It is available with either video output, and the image stream does not change at all when it is built. Off, the default, builds none of it.

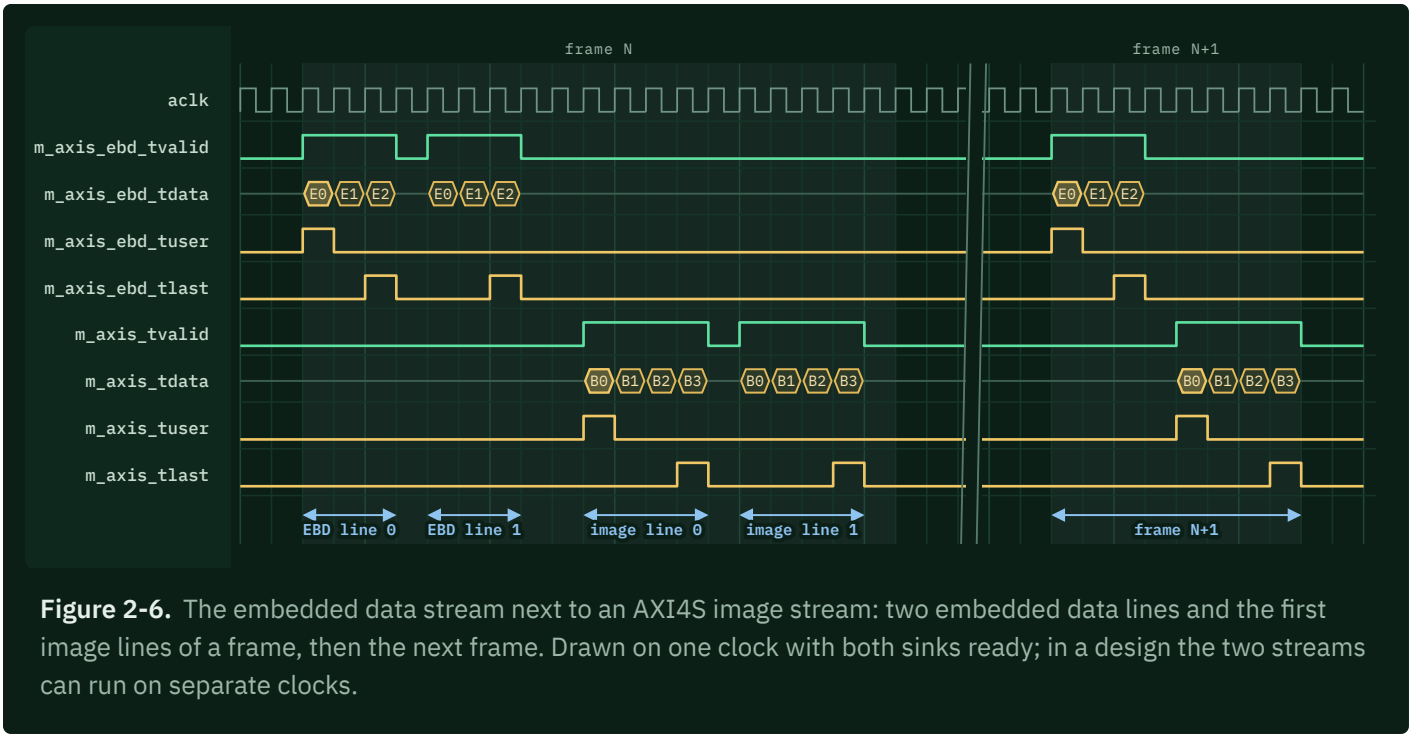


Figure 2-6. The embedded data stream next to an AXI4S image stream: two embedded data lines and the first image lines of a frame, then the next frame. Drawn on one clock with both sinks ready; in a design the two streams can run on separate clocks.

Table 2-7. Embedded data output at a glance.

Lines	Every packet with the embedded data line bit set, inside the frame or on the Frame End packet, from the first Frame Start after reset
Beat format	The image beat format: 2 x LANE_NUM 16-bit containers, MSB-justified at Pixel Bit
Line length	Each line is cut at Line Length: ceil(Line Length / (2 x LANE_NUM)) beats. The crop registers do not apply
Framing	`m_axis_ebd_tuser` on the first embedded data beat after each Frame Start, `m_axis_ebd_tlast` on the last beat of each line. Frames without embedded data send nothing
Pairing with the image	Count `tuser` on both streams: the *n*-th `tuser` on `M_AXIS_EBD` belongs to the frame that opened with the *n*-th `tuser` on the image stream, for a sensor that sends embedded data in every frame
Backpressure	Absorbed by the EBD FIFO. On overflow the line in progress ends with `tlast`, the rest of that frame's embedded data is dropped, and the stream restarts at the next Frame Start. The image stream never stalls for it
Control and status	`VID_CTRL.EBD_OFF` turns it off at run time; `EBD_FRAME_INFO` counts frames and lines; `EBD_OVF_CNT` reports overflows

A design that does not use the metadata builds without the output. A design that builds it but does not always read it ties `m_axis_ebd_tready` high or sets `VID_CTRL.EBD_OFF` . The content of the embedded data lines is defined by the sensor's datasheet.

© 2026 ApotheoTech LLC | apotheotech.com | info@apotheotech.com

2.8 Framing and crop

Both outputs deliver clean frames:

- One frame per Frame Start. Lines that arrive before the first Frame Start after reset are dropped, so the first thing out is always a whole frame.
- Image lines only. Embedded data lines go to the embedded data output when it is built; blanking packets are always removed.
- Every line is cut at Line Length, including the Frame End line, so stuffing and footer bytes never appear as pixels.
- Three crop registers remove lines and columns, applied from the next Frame Start.

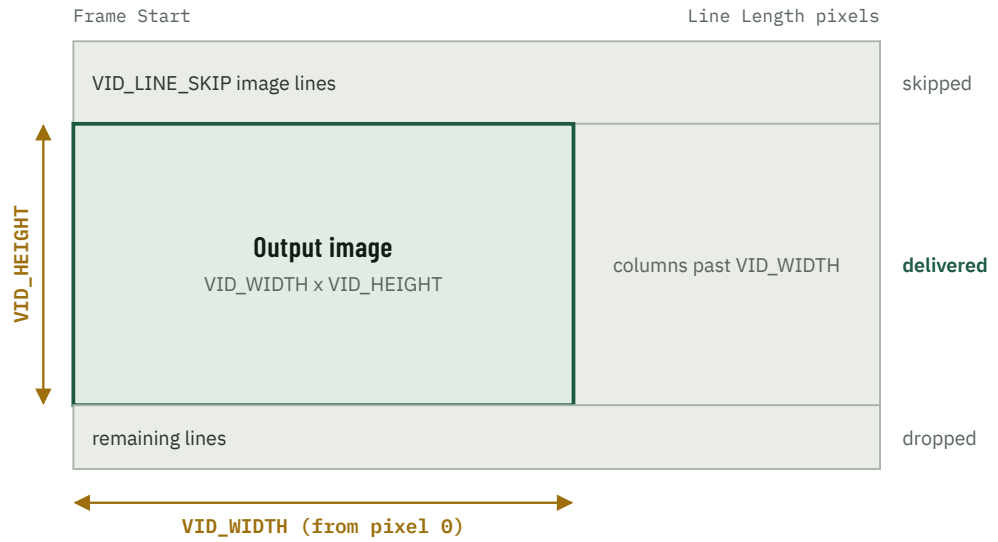


Figure 2-7. What reaches the output. The crop values are sampled at each Frame Start, so they can be changed while the sensor streams.

Table 2-8. Crop registers. All three reset to 0, which delivers the full image.

REGISTER	OFFSET	EFFECT
VID_LINE_SKIP	0x30	Image lines dropped at the start of every frame
VID_WIDTH	0x34	Pixels kept from pixel 0 of each line, rounded up to a whole beat of 2 x LANE_NUM pixels; 0 = whole line
VID_HEIGHT	0x38	Lines kept after the skip; 0 = all lines to Frame End

Example: an IMX530 window on 2 lanes, 5328 pixels wide, cropped to 1920 x 1080 starting at line 8: `VID_LINE_SKIP = 8`, `VID_WIDTH = 1920`, `VID_HEIGHT = 1080`, which gives 480 beats per line. The crop keeps the left part of the line; move the window horizontally with the sensor's own region-of-interest registers.

2.9 FIFO sizing

The core has one FIFO between the link and the video output, sized with **Output FIFO depth (words)** in the IP GUI: 128 to 16,384 words, default 2,048. One word is one output beat of 2 x LANE_NUM pixels. The embedded data output, when built, has a second FIFO of its own.

© 2026 ApotheoTech LLC | apotheotech.com | info@apotheotech.com

2.9.1 NATIVE: the longest line

Native video replays each line in one burst once it has arrived, so the buffer must hold a whole line. 32 words are reserved for flow control.

```
beats per line  = Line Length / (2 x LANE_NUM)        (after VID_WIDTH)
requirement     : beats per line <= depth - 32
longest line    = (depth - 32) x 2 x LANE_NUM pixels
```

Table 2-9. NATIVE output: longest line in pixels for each Output FIFO depth (words) and lane count, (depth - 32) x 2 x LANE_NUM.

DEPTH	1 LANE	2 LANES	4 LANES	6 LANES	8 LANES
128	192	384	768	1,152	1,536
256	448	896	1,792	2,688	3,584
512	960	1,920	3,840	5,760	7,680
1,024	1,984	3,968	7,936	11,904	15,872
2,048 (default)	4,032	8,064	16,128	24,192	32,256
4,096	8,128	16,256	32,512	48,768	65,024
8,192	16,320	32,640	65,280	97,920	130,560
16,384	32,704	65,408	130,816	196,224	261,632

A full-width IMX530 line (5328 pixels) needs 1,332 beats on 2 lanes, so 2,048 words; on 8 lanes it needs 333 beats and 512 words are enough. 1920-pixel lines on 2 lanes fit exactly in 512 words; 1,024 leaves room for other modes.

2.9.2 AXI4S: how long the sink may stall

The AXI4-Stream FIFO never has to hold a line. It sets how long `m_axis_tready` may stay low before a frame is cut. The lane count drops out: more lanes deliver more pixels per clock, but each word holds more pixels too.

```
stall budget = (depth - 8) x Pixel Bit / (8 x f_lane_clock)
```

Table 2-10. AXI4S output: longest m_axis_tready stall in microseconds before a frame is cut, at a 237.6 MHz lane clock (4.752 Gb/s lanes). Independent of the lane count. Scale by 237.6 MHz / your lane clock.

DEPTH	RAW8	RAW10	RAW12	RAW14	RAW16
128	0.51	0.63	0.76	0.88	1.0
256	1.0	1.3	1.6	1.8	2.1
512	2.1	2.7	3.2	3.7	4.2
1,024	4.3	5.3	6.4	7.5	8.6
2,048 (default)	8.6	10.7	12.9	15.0	17.2

© 2026 ApotheoTech LLC | apotheotech.com | info@apotheotech.com

DEPTH	RAW8	RAW10	RAW12	RAW14	RAW16
4,096	17.2	21.5	25.8	30.1	34.4
8,192	34.4	43.1	51.7	60.3	68.9
16,384	68.9	86.2	103	121	138

A VDMA or a pipeline that is always ready needs little buffering: 512 to 2,048 words is comfortable. A sink that stalls for a known time, such as a frame buffer writer behind a busy memory controller, gets that time plus margin. A deeper FIFO costs block RAM only; it is first-word fall-through and adds no latency while `tready` is high. The stream clock only has to carry the average beat rate of the video; within a line the FIFO smooths the difference.

2.9.3 Embedded data FIFO

Sized with **EBD FIFO depth (words)**: 32 to 16,384 words, default 1,024. It holds the embedded data lines of one frame while the sink is busy. A sink that reads them only in the frame blanking needs

```
EBD FIFO depth >= E x ceil(Line Length / (2 x LANE_NUM))      E = embedded data lines per frame
```

Two embedded data lines of 3,840 pixels on 4 lanes are 960 words, so 1,024. A sink that keeps up while the lines arrive needs only the minimum of 32.

2.10 Overflow and recovery

A sensor cannot be paused. When the output FIFO fills, the core gives up part of the video in a way that keeps the output well formed, and recovers by itself. Nothing has to be reset.

Table 2-11. Overflow behaviour.

	NATIVE	AXI4S
Cause	A line longer than the line buffer	`m_axis_tready` low for longer than the FIFO absorbs
What happens	The line ends early as a short line; the rest of the frame is dropped; `vid_vblank` rises normally	The line in progress ends on its next beat with `tlast`; the rest of the frame is dropped; beats already in the FIFO come out in order
Recovery	Next Frame Start (lasting recovery needs lines that fit: crop or a deeper FIFO)	First Frame Start that finds room; the stream restarts with `tuser`
Reported	`VID_OVF_CNT`: bit 31 sticky flag, bits 15:0 frames cut or dropped. Cleared by reset or `VID_CTRL`	

The embedded data output behaves the same way on its own FIFO, reports in `EBD_OVF_CNT`, and never affects the image: a stalled metadata sink costs metadata only.

The consumer of an AXI4S output owns `m_axis_aresetn` (and `m_axis_ebd_aresetn` for the embedded data stream) and may pulse it at any time, for any length. `tvalid` drops at once, the FIFO is flushed and held empty while the reset is low, and the stream restarts on the next Frame Start with `tuser`. A stream reset is never counted as an overflow.

2.11 Clocks and reset

Table 2-12. Clock domains.

© 2026 ApotheoTech LLC | apotheotech.com | info@apotheotech.com

CLOCK	SOURCE	RUNS
`data_clk_i`	Lane clock from the bundled PHY: line rate / 20 (237.6 MHz at 4.752 Gb/s, 250 MHz at 5.0 Gb/s)	The link layer, the NATIVE output and the write side of the AXI4S FIFO
`axi_aclk`	Any free-running clock, typically the processor's fabric clock	The register file only
`m_axis_aclk`	The clock of the block that takes the stream, any frequency (AXI4S build)	The read side of the AXI4S FIFO and `M_AXIS`
`m_axis_ebd_aclk`	The clock of the block that takes the embedded data, any frequency; may be the same clock as `m_axis_aclk` (embedded data output built)	The read side of the EBD FIFO and `M_AXIS_EBD`

- Hold `rstn_i` low until the PHY reports the lanes ready. The example designs show the connection.
- Clock `axi_aclk` from a free-running clock rather than the lane clock. The registers then stay readable while the link is down, and software can read ID and VERSION at boot before the sensor starts.
- Every crossing between these clocks is inside the core and constrains itself. Do not add `set_clock_groups` or blanket false paths between these clocks: they would override the core's own constraints.
- After reset the output stage needs a short settling time (about 40 lane clocks for NATIVE, 100 clocks of the slower clock for AXI4S). A frame that starts during that time is skipped, so a streaming sensor simply loses its first frame.

© 2026 ApotheoTech LLC | apotheotech.com | info@apotheotech.com

SECTION 03

Interfaces and Timing

IP parameters, every port of the core, and the timing of the AXI4S, NATIVE and embedded data outputs.

3.1 IP customization parameters

All parameters are set when the IP is generated in Vivado. Software reads them back from the `HWCFG` register, so one driver serves every build.

Table 3-1. IP customization parameters.

GUI FIELD	PARAMETER	VALUES	DEFAULT	DESCRIPTION
Lanes	LANE_NUM	1, 2, 4, 6, 8	4	SLVS-EC lane count; must match the sensor's Lane Num.
Payload protection	ECC_SEL	CRC (0), ECC Reed-Solomon (1)	CRC	Builds the CRC32 checker or the Reed-Solomon corrector; must match the sensor mode.
Output	OUT_MODE	AXI4S (1), NATIVE (0)	NATIVE	Video interface; only the ports of the selected output are shown.
Output FIFO depth (words)	OUT_FIFO_DEPTH	128 to 16384, powers of 2	2048	NATIVE line buffer or AXI4S FIFO.
Embedded data output	EBD_OUT	Off (0), On (1)	Off	Builds the embedded data stream `M_AXIS_EBD` with its clock, reset and FIFO, next to either video output.
EBD FIFO depth (words)	EBD_FIFO_DEPTH	32 to 16384, powers of 2	1024	Embedded data FIFO, shown with Embedded data output on.

From a block design script:

```
set_property -dict [list \
    CONFIG.LANE_NUM       {2} \
    CONFIG.ECC_SEL        {0} \
    CONFIG.OUT_MODE       {1} \
    CONFIG.OUT_FIFO_DEPTH {2048} \
    CONFIG.EBD_OUT        {1} \
    CONFIG.EBD_FIFO_DEPTH {1024} \
] [get_bd_cells slvs_ec_rx_0]
```

The PHY is delivered set up for your sensor mode and board; there is no transceiver configuration to do.

© 2026 ApotheoTech LLC | apotheotech.com | info@apotheotech.com

3.2 Interface overview

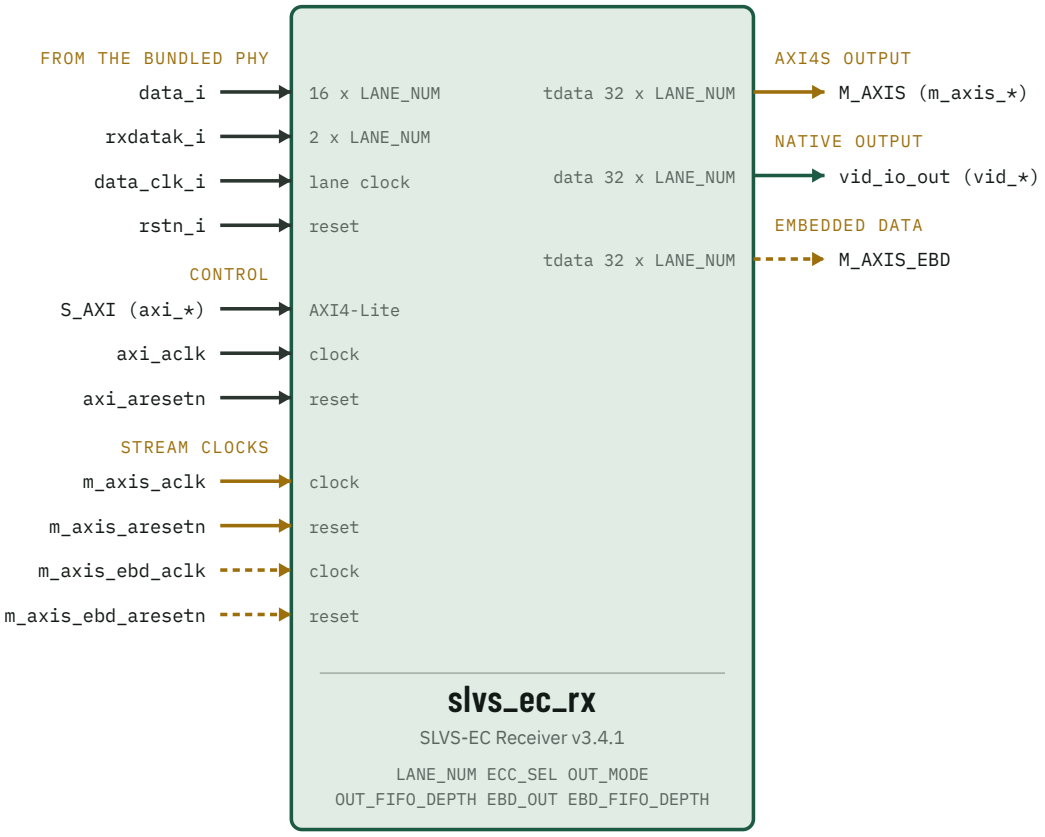


Figure 3-1. IP symbol. Dark ports exist in every build, gold ports in the AXI4S build, green ports in the NATIVE build. Dashed ports appear with Embedded data output on.

3.3 Port descriptions

3.3.1 PHY connection

Table 3-2. Ports between the core and the bundled PHY (APT-SLVS-EC GT Wizard or SLVS-EC HP I/O). Connect them as in the example designs.

PORT	DIR	WIDTH	DESCRIPTION
data_i	in	16 x LANE_NUM	Decoded lane data from the PHY
rxdatak_i	in	2 x LANE_NUM	Control-character flags from the PHY
data_clk_i	in	1	Lane clock from the PHY, line rate / 20
rstn_i	in	1	Active-low synchronous reset on `data_clk_i`. Hold low until the PHY reports the lanes ready

© 2026 ApotheoTech LLC | apotheotech.com | info@apotheotech.com

3.3.2 AXI4-Lite control

Table 3-3. Register interface S_AXI. Present in every build.

PORT	DIR	WIDTH	DESCRIPTION
axi_aclk	in	1	Register clock, independent of every other clock
axi_aresetn	in	1	Active-low synchronous reset on `axi_aclk`; resets every RW register to 0
axi_aw* / axi_w* / axi_b*	in/out	addr 8, data 32	Write channels. `WSTRB` is honoured; `BRESP` is always OKAY; `AWPROT` is ignored
axi_ar* / axi_r*	in/out	addr 8, data 32	Read channels. `RRESP` is always OKAY; unused offsets read 0

One write and one read are in flight at a time, and `BREADY` or `RREADY` backpressure never loses a response. The IP reserves a 4 KB address range in the Vivado address editor; the registers occupy the first 256 bytes.

3.3.3 AXI4S output (Output = AXI4S)

Table 3-4. AXI4-Stream video master M_AXIS.

PORT	DIR	WIDTH	DESCRIPTION
m_axis_aclk	in	1	Stream clock, any frequency, no phase relation to `data_clk_i`
m_axis_aresetn	in	1	Active-low synchronous reset on `m_axis_aclk`; may be pulsed at any time
m_axis_tdata	out	32 x LANE_NUM	2 x LANE_NUM pixels in 16-bit MSB-justified containers, earliest pixel in bits 15:0
m_axis_tvalid	out	1	Held with the payload until `tready`; low during reset
m_axis_tready	in	1	May change on any clock
m_axis_tuser	out	1	Start of frame: first beat of the first line of a frame
m_axis_tlast	out	1	End of line: last beat of every line

There is no `tkeep` or `tstrb`: every beat is a full beat. Put an AXI4-Stream data width converter or subset converter behind the core when the sink wants a different number of pixels per beat or 8-bit components.

3.3.4 NATIVE output (Output = NATIVE)

Table 3-5. Native video master vid_io_out, on data_clk_i.

PORT	DIR	WIDTH	DESCRIPTION
vid_data	out	32 x LANE_NUM	2 x LANE_NUM pixels per clock, same format as `m_axis_tdata`
vid_active_video	out	1	High for every beat of a line, no gaps inside a line
vid_hblank	out	1	High outside a line; the inverse of `vid_active_video`

© 2026 ApotheoTech LLC | apotheotech.com | info@apotheotech.com

PORT	DIR	WIDTH	DESCRIPTION
vid_vblank	out	1	High between frames
vid_hsync	out	1	1-clock pulse at the start of horizontal blanking after each line
vid_vsync	out	1	4-clock pulse at the start of vertical blanking after each frame
vid_field_id	out	1	Always 0, progressive

3.3.5 Embedded data output (Embedded data output = On)

Table 3-6. AXI4-Stream master M_AXIS_EBD. Available with either video output.

PORT	DIR	WIDTH	DESCRIPTION
m_axis_ebd_aclk	in	1	Embedded data stream clock, any frequency, no phase relation to `data_clk_i`; may be the same clock as `m_axis_aclk`
m_axis_ebd_aresetn	in	1	Active-low synchronous reset on `m_axis_ebd_aclk`; may be pulsed at any time
m_axis_ebd_tdata	out	32 x LANE_NUM	2 x LANE_NUM 16-bit words, MSB-justified at Pixel Bit: the image beat format
m_axis_ebd_tvalid	out	1	Held with the payload until `tready`; low during reset
m_axis_ebd_tready	in	1	May change on any clock. Tie high if the metadata is not read
m_axis_ebd_tuser	out	1	First embedded data beat after a Frame Start: one frame's embedded data starts here
m_axis_ebd_tlast	out	1	Last beat of every embedded data line

3.4 Output timing

3.4.1 AXI4S

Standard AXI4-Stream rules: `tvalid` and the payload hold until the handshake, and `tready` may do anything while the FIFO has room. The capture below shows the start of a frame, two line ends and a sink stall.

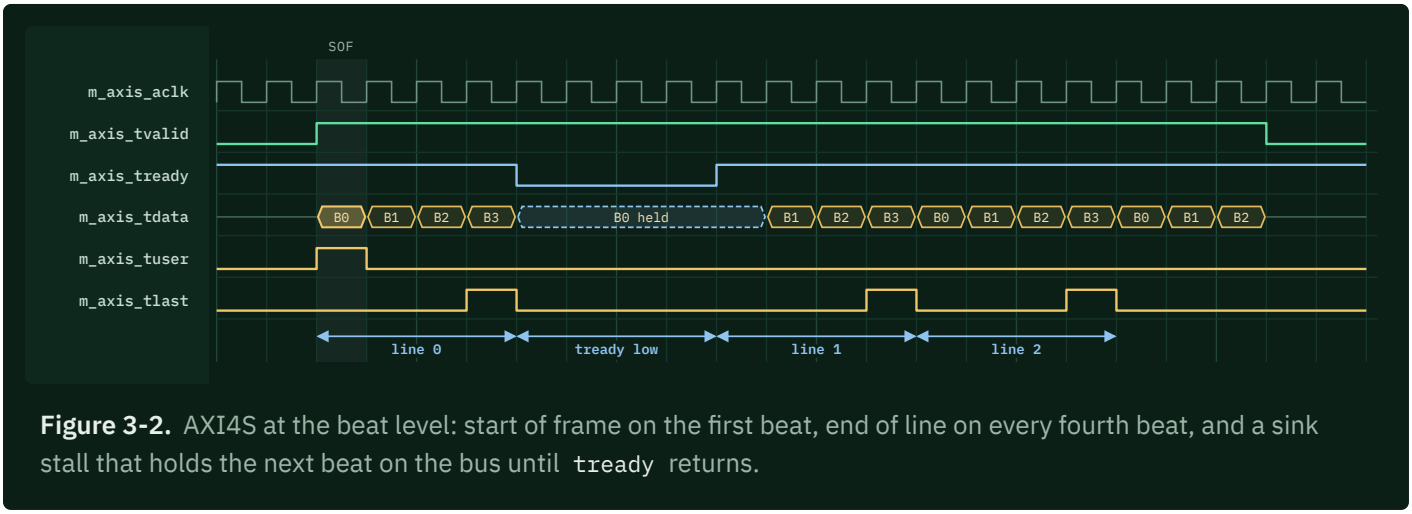


Figure 3-2. AXI4S at the beat level: start of frame on the first beat, end of line on every fourth beat, and a sink stall that holds the next beat on the bus until `tready` returns.

3.4.2 NATIVE

The blanking follows the sensor: the line pitch on the output is the line pitch on the link. The capture shows the shortest intervals the output can produce, reached only when lines or frames arrive back to back.

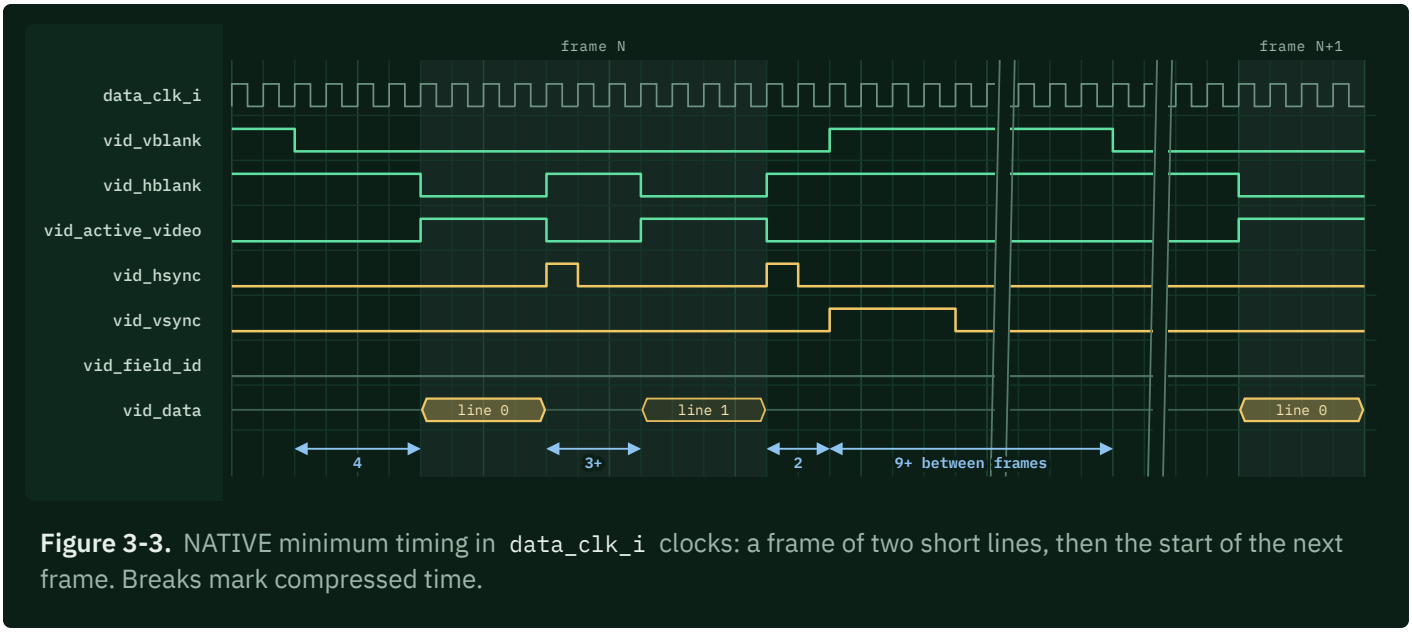


Figure 3-3. NATIVE minimum timing in `data_clk_i` clocks: a frame of two short lines, then the start of the next frame. Breaks mark compressed time.

Table 3-7. NATIVE minimum timing.

INTERVAL	CLOCKS
`vid_vblank` fall to first beat of a frame	4
`vid_hblank` between two lines	3 or more
Last beat of a frame to `vid_vblank` rise	2
`vid_vblank` between two frames	9 or more
`vid_hsync` width / `vid_vsync` width	1 / 4
Latency, link to output	one line

To feed AMD Video In to AXI4-Stream, connect `vid_io_out` to `vid_io_in`, clock `vid_io_in_clk` from `data_clk_i` with `vid_io_in_ce` tied high, and set its width for 32 x LANE_NUM bits (for example 4 pixels per clock, 1 component of 16 bits for 2 lanes). Use its independent clock mode to move the stream to another clock. A block built for frame, line and data valid inputs takes fv = not `vid_vblank`, lv = dv = `vid_active_video`. The NATIVE output never carries embedded data lines; with the embedded data output built they arrive on `M_AXIS_EBD`.

3.4.3 Embedded data

The same AXI4-Stream rules as the image stream, on its own clock and FIFO. Each embedded data line is ceil(Line Length / (2 x LANE_NUM)) beats long and ends with `tlast`; the first beat after each Frame Start carries `tuser`. A frame without embedded data lines sends nothing, so the stream is idle between groups. The capture in Embedded data output shows it next to the image stream.

© 2026 ApotheoTech LLC | apotheotech.com | info@apotheotech.com

3.5 Integration notes

- **Configuration is quasi-static.** `PIX_BIT`, `LINE_LENGTH`, `ECC_OPTION`, `CRC_OPTION` and `ECC_BLOCK` reach the link as one word, but not aligned to a packet. Write them while the sensor is idle. The crop registers are sampled at Frame Start and may be written at any time.
- **Line Length.** Program the sensor's pixels per line: a multiple of 4 at 1, 2, 4 and 6 lanes, a multiple of 8 at 8 lanes. A Line Length that is not a multiple of 2 x LANE_NUM ends on a partial beat whose upper containers carry non-image data.
- **Payload protection.** Keep CRC Option on in the CRC build and ECC Option 1 or 2 in the ECC build; the sensor must be set the same way. The driver warns at stream start when it is off.
- **ECC_BLOCK.** Keep it programmed in the ECC build even when the sensor carries the line length in the packet header.
- **NATIVE line buffer.** Size the Output FIFO so Line Length / (2 x LANE_NUM) is at most depth - 32 beats after the crop.
- **Inter-packet codes.** The sensor must send Idle, Pad or Deskew Codes between packets as the standard requires; all Sony SLVS-EC sensors do. At 8 lanes a packet whose End Code runs straight into the next Start Code is lost.

© 2026 ApotheoTech LLC | apotheotech.com | info@apotheotech.com

SECTION 04

Register Map

Twenty-four 32-bit registers on AXI4-Lite: identification, link configuration, error counters, video output and embedded data output, with bit fields, reset values and the programming sequence.

4.1 Register interface

- AXI4-Lite slave, 32-bit data, word access. Offsets below are byte offsets from the base address assigned in the Vivado address editor.
- Writes honour `WSTRB`. Writes to read-only registers and unused offsets are ignored. Every access returns OKAY, so software can probe any offset without a bus fault.
- `axi_aresetn` resets every read/write register to 0. `PIX_BIT` and `LINE_LENGTH` must therefore be programmed before the sensor streams.
- Reserved bits of read/write registers are stored and read back; write them as 0.
- The first four words (ID, VERSION, HWCFG, SCRATCH) follow the same layout in every ApotheoTech core, so one probe routine identifies any of them.

4.2 Register summary

Table 4-1. Register summary. Offsets are byte offsets from the core base address. Offsets 0x60 to 0xFC read 0 and ignore writes.

OFFSET	NAME	ACCESS	RESET	FUNCTION
IDENTIFICATION				
0x00	ID	RO	0xA715EC01	Core identification word
0x04	VERSION	RO	0x03040001	IP release, major.minor.patch
0x08	HWCFG	RO	build	Build parameters of this instance
0x0C	SCRATCH	RW	0x00000000	Software scratch word
LINK CONFIGURATION				
0x10	PIX_BIT	RW	0x00000000	Pixel bit depth
0x14	LINE_LENGTH	RW	0x00000000	Line length in pixels
0x18	ECC_OPTION	RW	0x00000000	Reed-Solomon ECC option (ECC build)
0x1C	CRC_OPTION	RW	0x00000000	Payload CRC32 check (CRC build)
0x20	ECC_BLOCK	RW	0x00000000	ECC block layout of a line (ECC build)
ERROR COUNTERS				
0x24	ERR_CNT_CTRL	RW	0x00000000	Error counter control

© 2026 ApotheoTech LLC | apotheotech.com | info@apotheotech.com

OFFSET	NAME	ACCESS	RESET	FUNCTION
0x28	C_ERR_CNT	RO	0x00000000	Corrected symbols, running total (ECC build)
0x2C	FRAME_ERR_CNT	RO	0x00000000	Corrected symbols in the current frame (ECC build)
0x48	ECC_FAIL_CNT	RO	0x00000000	Uncorrectable blocks, running total (ECC build)
0x4C	HDR_CRC_ERR_CNT	RO	0x00000000	Header CRC16 failures: bad copies and lost headers
0x50	PAY_CRC_ERR_CNT	RO	0x00000000	Payload CRC32 failures (CRC build)
0x54	CRC_ERR_LINE	RO	0x00000000	Last line with a CRC error, and a sticky error flag
VIDEO OUTPUT				
0x30	VID_LINE_SKIP	RW	0x00000000	Image lines dropped at the top of every frame
0x34	VID_WIDTH	RW	0x00000000	Pixels kept per line
0x38	VID_HEIGHT	RW	0x00000000	Lines kept per frame
0x3C	VID_CTRL	RW	0x00000000	Video and embedded data output control
0x40	VID_FRAME_INFO	RO	0x00000000	Frame counter and delivered height
0x44	VID_OVF_CNT	RO	0x00000000	Output FIFO overflow status
EMBEDDED DATA OUTPUT				
0x58	EBD_FRAME_INFO	RO	0x00000000	Embedded data frame and line counts
0x5C	EBD_OVF_CNT	RO	0x00000000	Embedded data FIFO overflow status

Read-only Read/write Reserved

4.3 Identification

Read ID and VERSION before any write: they confirm the base address and the release. HWCFG tells software how the IP was built, so one driver handles CRC and ECC builds, every lane count and every output.

0x00 ID RO

reset 0xA715EC01

Fixed identification word. Software reads it first to confirm the base address before any write.



BITS	FIELD	ACCESS	RESET	DESCRIPTION
[31:20]	VENDOR	RO	0xA71	Vendor code: 0xA71 = ApotheoTech.
[19:8]	CORE	RO	0x5EC	Core code: 0x5EC = SLVS-EC.
[7:0]	FUNCTION	RO	0x01	Function: 0x01 = receiver (RX).

© 2026 ApotheoTech LLC | apotheotech.com | info@apotheotech.com

0x04 VERSION RO

reset 0x03040001

Release of the IP in the bitstream. A new major number means a register map change; a new minor number is a compatible addition; a patch has no software-visible change.



BITS	FIELD	ACCESS	RESET	DESCRIPTION
[31:24]	MAJOR	RO	0x03	Major version.
[23:16]	MINOR	RO	0x04	Minor version.
[15:0]	PATCH	RO	0x0001	Patch level.

0x08 HWCFG RO

reset build

Reports the parameters the IP was generated with, so one software build can serve several hardware builds. Example: 4 lanes, CRC build, AXI4S output with a 2048-word FIFO, embedded data output on with a 1024-word FIFO reads 0x0A0B0A04.



BITS	FIELD	ACCESS	RESET	DESCRIPTION
[31:29]	reserved		0	Reserved, reads 0.
[28:24]	EBD_FIFO_AW	RO	build	Embedded data FIFO depth as log2 of words: depth = 1 << EBD_FIFO_AW. 10 = 1024 words. 0 when the embedded data output is not built.
[23:21]	reserved		0	Reserved, reads 0.
[20:16]	OUT_FIFO_AW	RO	build	Output FIFO depth as log2 of words: depth = 1 << OUT_FIFO_AW. 11 = 2048 words.
[15:12]	reserved		0	Reserved, reads 0.
[11]	EBD_OUT	RO	build	1 = the embedded data output M_AXIS_EBD is built.
[10:9]	OUT_MODE	RO	build	Video output: 0 = NATIVE, 1 = AXI4S.
[8]	ECC_SEL	RO	build	Payload protection: 0 = CRC build, 1 = ECC build (Reed-Solomon).
[7:4]	reserved		0	Reserved, reads 0.
[3:0]	LANE_NUM	RO	build	Lane count: 1, 2, 4, 6 or 8.

© 2026 ApotheoTech LLC | apotheotech.com | info@apotheotech.com

0x0C SCRATCH RW

reset 0x00000000

Free for software. No effect on the core.



BITS	FIELD	ACCESS	RESET	DESCRIPTION
[31:0]	SCRATCH	RW	0	Any value.

4.4 Link configuration

These registers mirror the SLVS-EC Configuration Register of the sensor. They are quasi-static: write them while the sensor is idle, before it starts streaming.

0x10 PIX_BIT RW

reset 0x00000000

Pixel Bit of the sensor mode (SLVS-EC Configuration Register). Must be programmed before the sensor streams.

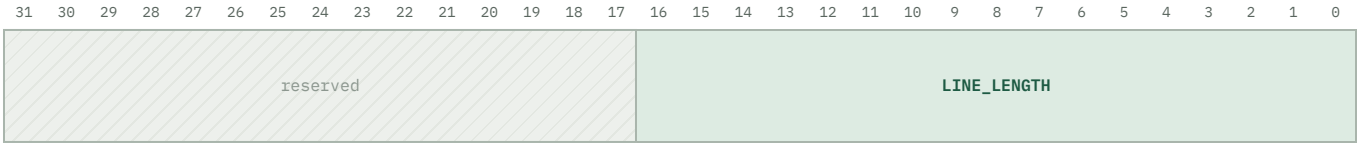


BITS	FIELD	ACCESS	RESET	DESCRIPTION
[31:5]	reserved		0	Reserved, write 0.
[4:0]	PIX_BIT	RW	0	Bits per pixel: 8, 10, 12, 14 or 16.

0x14 LINE_LENGTH RW

reset 0x00000000

Line Length of the sensor mode, in pixels: the sensor's pixels per line, for image lines and embedded data lines alike. A multiple of 4 at 1, 2, 4 and 6 lanes, a multiple of 8 at 8 lanes. The core cuts every line at this length, so payload stuffing never reaches an output. When the sensor carries the line length in the packet header, the core uses the header value.



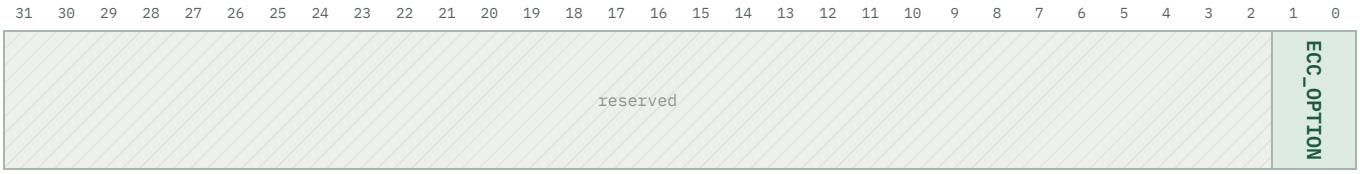
BITS	FIELD	ACCESS	RESET	DESCRIPTION
[31:17]	reserved		0	Reserved, write 0.
[16:0]	LINE_LENGTH	RW	0	Pixels per line. Multiple of 4 (1, 2, 4, 6 lanes) or 8 (8 lanes). The core uses bits [15:0].

© 2026 ApotheoTech LLC | apotheotech.com | info@apotheotech.com

0x18 ECC_OPTION RW

reset 0x00000000

Selects the payload ECC the sensor uses. Used by the ECC build, which keeps Option 1 or 2 on while an image streams; the supplied driver warns when it is off. Writing 0 also clears the three ECC counters.

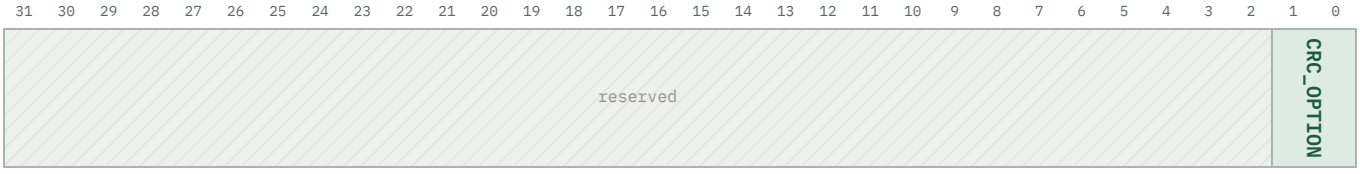


BITS	FIELD	ACCESS	RESET	DESCRIPTION
[31:2]	reserved		0	Reserved, write 0.
[1:0]	ECC_OPTION	RW	0	00 = off; 01 = Option 1, 2 parity bytes per block, corrects 1 byte; 10 = Option 2, 4 parity bytes per block, corrects up to 2 bytes; 11 = reserved.

0x1C CRC_OPTION RW

reset 0x00000000

Enables the payload CRC32 footer check when the sensor appends the footer. Used by the CRC build, which keeps it on while an image streams; the supplied driver warns when it is off.

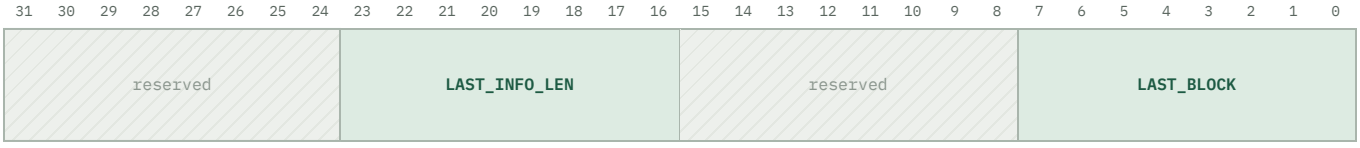


BITS	FIELD	ACCESS	RESET	DESCRIPTION
[31:2]	reserved		0	Reserved, write 0.
[1:0]	CRC_OPTION	RW	0	00 = off; any other value = check the CRC32 footer of every packet.

0x20 ECC_BLOCK RW

reset 0x00000000

Where the last Reed-Solomon block of a line starts. Computed from Line Length and Pixel Bit; see Programming ECC_BLOCK. The supplied driver computes it for you.



BITS	FIELD	ACCESS	RESET	DESCRIPTION
[31:24]	reserved		0	Reserved, write 0.
[23:16]	LAST_INFO_LEN	RW	0	Information length of the last block in bytes, multiple of 16 (16 to 224).
[15:8]	reserved		0	Reserved, write 0.
[7:0]	LAST_BLOCK	RW	0	Index of the last block of the line.

© 2026 ApotheoTech LLC | apotheotech.com | info@apotheotech.com

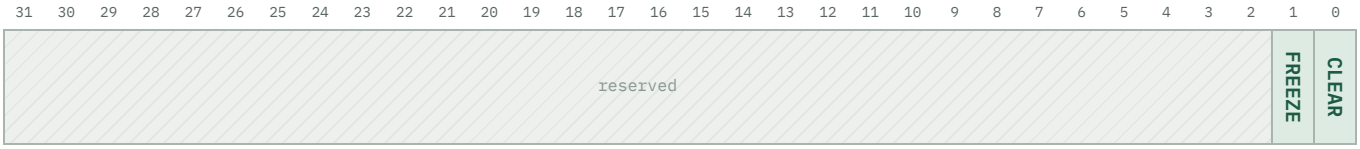
4.5 Error counters

Header CRC16 failures are counted in both builds, as bad copies and as lost headers. The CRC build counts payload CRC32 failures and keeps the Line Number of the last failing line; the ECC build counts corrected bytes per frame and in total, and uncorrectable blocks in total. Counters that belong to the other build read 0. `CRC_ERR_LINE[31]` is one sticky bit that says whether anything went wrong since the last clear.

0x24 ERR_CNT_CTRL RW

reset 0x00000000

Controls the running error counters C_ERR_CNT and ECC_FAIL_CNT and the CRC error registers HDR_CRC_ERR_CNT, PAY_CRC_ERR_CNT and CRC_ERR_LINE. FRAME_ERR_CNT is cleared by the core at every Frame Start. The CRC error registers are cleared only by this bit and the core reset.

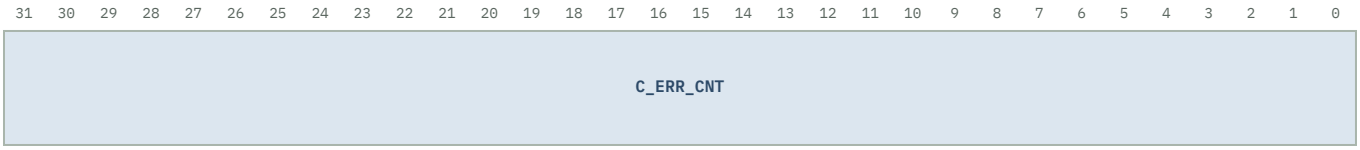


BITS	FIELD	ACCESS	RESET	DESCRIPTION
[31:2]	reserved		0	Reserved, write 0.
[1]	FREEZE	RW	0	1 = hold the running error counters at their current values.
[0]	CLEAR	RW	0	1 = clear the running error counters, held at 0 while the bit is 1. Write 1 then 0.

0x28 C_ERR_CNT RO

reset 0x00000000

Bytes corrected by the Reed-Solomon decoder since the last clear. A slow count is the ECC doing its job; a fast count points at signal integrity. Reads 0 in the CRC build.

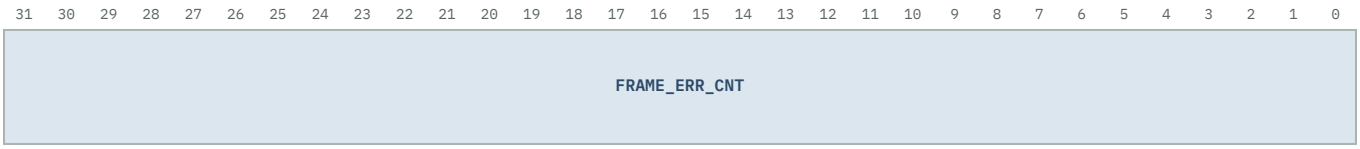


BITS	FIELD	ACCESS	RESET	DESCRIPTION
[31:0]	C_ERR_CNT	RO	0	Corrected symbols.

0x2C FRAME_ERR_CNT RO

reset 0x00000000

Bytes corrected since the last Frame Start. Reads 0 in the CRC build.



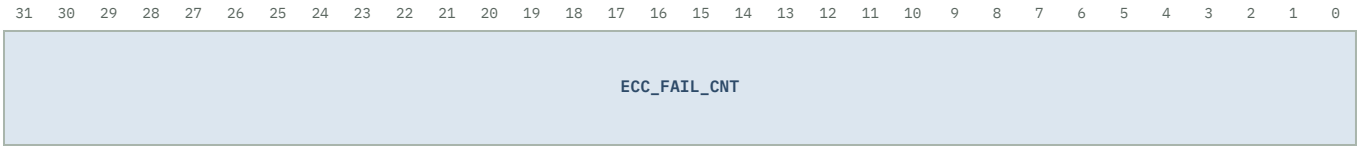
BITS	FIELD	ACCESS	RESET	DESCRIPTION
[31:0]	FRAME_ERR_CNT	RO	0	Corrected symbols since the last Frame Start.

© 2026 ApotheoTech LLC | apotheotech.com | info@apotheotech.com

0x48 ECC_FAIL_CNT RO

reset 0x00000000

Reed-Solomon blocks with more errors than the selected option corrects. Such a block is passed on unchanged and counted here; on a healthy link this stays at 0. Reads 0 in the CRC build.



BITS	FIELD	ACCESS	RESET	DESCRIPTION
[31:0]	ECC_FAIL_CNT	RO	0	Uncorrectable blocks.

0x4C HDR_CRC_ERR_CNT RO

reset 0x00000000

Every packet header is sent three times, each copy with its own CRC16. The core checks a packet once its third copy is in. One or two bad copies are recovered from a good one and point at a marginal link; a lost header (all three copies bad) loses the packet. Lost headers at bring-up almost always mean lane order, byte order or deskew. Counts in both builds; each field stops at 0xFFFF.

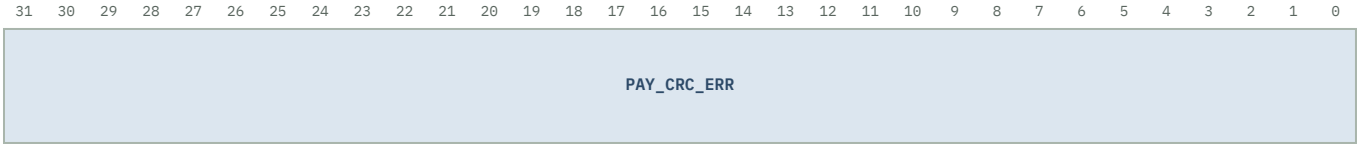


BITS	FIELD	ACCESS	RESET	DESCRIPTION
[31:16]	HDR_LOST	RO	0	Headers lost: all three copies failed CRC16.
[15:0]	COPY_ERR	RO	0	Header copies that failed CRC16, 0 to 3 per packet.

0x50 PAY_CRC_ERR_CNT RO

reset 0x00000000

Packets whose CRC32 footer does not match their payload, counted with CRC_OPTION on. The pixels are passed on as received. The ECC build has no CRC32 footer: the register reads 0 there and ECC_FAIL_CNT counts uncorrectable blocks instead. Stops at 0xFFFFFFFF.



BITS	FIELD	ACCESS	RESET	DESCRIPTION
[31:0]	PAY_CRC_ERR	RO	0	Packets with a payload CRC32 mismatch.

© 2026 ApotheoTech LLC | apotheotech.com | info@apotheotech.com

0x54 CRC_ERR_LINE RO

reset 0x00000000

Bit 31 sets with the first lost header or payload CRC32 error after a clear, one bit for software to poll. Bits [12:0] take the Line Number of every packet with a payload CRC32 error, so they hold the last one. A packet whose header was lost as well has no Line Number: it is counted in PAY_CRC_ERR_CNT and leaves the line field as it was.



BITS	FIELD	ACCESS	RESET	DESCRIPTION
[31]	ERR_SEEN	RO	0	1 = a lost header or a payload CRC32 error since the last clear.
[30:13]	reserved		0	Reserved, reads 0.
[12:0]	LINE_NUM	RO	0	Line Number of the last packet with a payload CRC32 error.

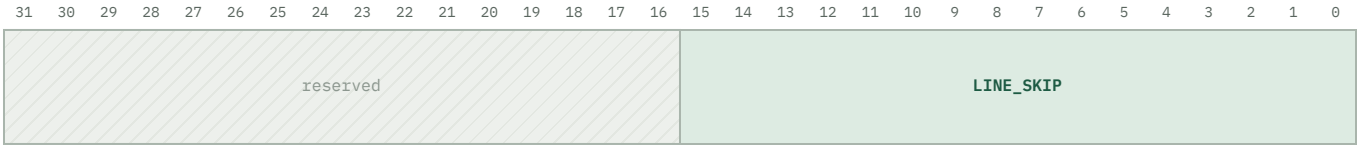
4.6 Video output

These registers control and report the video output stage: the crop, the frame counter and the output overflow status. `VID_CTRL` also holds the two embedded data output controls.

0x30 VID_LINE_SKIP RW

reset 0x00000000

Crop: number of image lines removed at the start of every frame. Taken at each Frame Start, so it can be written while streaming.

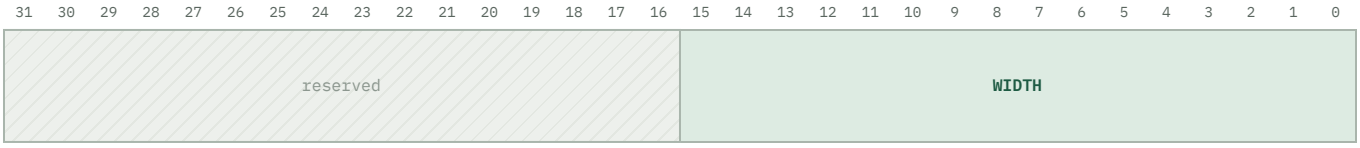


BITS	FIELD	ACCESS	RESET	DESCRIPTION
[31:16]	reserved		0	Reserved, write 0.
[15:0]	LINE_SKIP	RW	0	Lines to skip. 0 = none.

0x34 VID_WIDTH RW

reset 0x00000000

Crop: pixels kept from the start of each line. Rounded up to a whole beat of 2 x LANE_NUM pixels. Taken at each Frame Start.



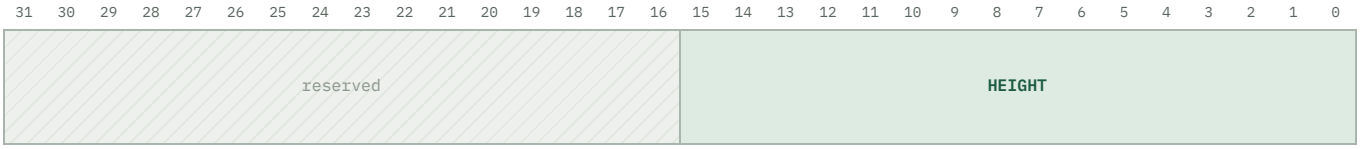
BITS	FIELD	ACCESS	RESET	DESCRIPTION
[31:16]	reserved		0	Reserved, write 0.
[15:0]	WIDTH	RW	0	Pixels per line. 0 = the whole line.

© 2026 ApotheoTech LLC | apotheotech.com | info@apotheotech.com

0x38 VID_HEIGHT RW

reset 0x00000000

Crop: image lines kept after the skip. Taken at each Frame Start.

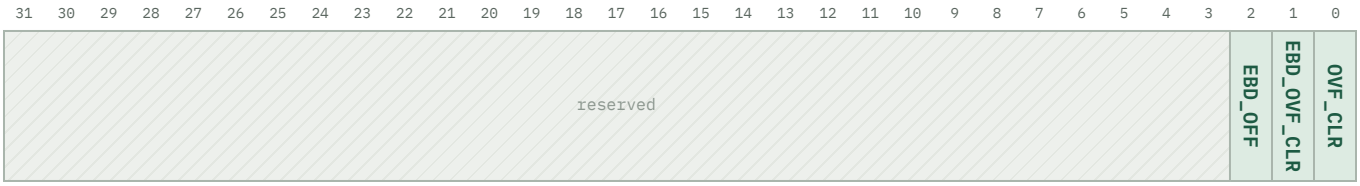


BITS	FIELD	ACCESS	RESET	DESCRIPTION
[31:16]	reserved		0	Reserved, write 0.
[15:0]	HEIGHT	RW	0	Lines per frame. 0 = every line up to Frame End.

0x3C VID_CTRL RW

reset 0x00000000

Clears the overflow status of the two outputs and turns the embedded data output off.



BITS	FIELD	ACCESS	RESET	DESCRIPTION
[31:3]	reserved		0	Reserved, write 0.
[2]	EBD_OFF	RW	0	1 = drop embedded data lines; the image does not change. Sampled between embedded data lines, so a line on its way is finished.
[1]	EBD_OVF_CLR	RW	0	While 1, the EBD overflow flag and EBD_OVF_CNT are held at 0. Write 1 then 0 to clear.
[0]	OVF_CLR	RW	0	While 1, the image overflow flag and VID_OVF_CNT are held at 0. Write 1 then 0 to clear.

0x40 VID_FRAME_INFO RO

reset 0x00000000

One read answers the first bring-up questions: are frames arriving, and how many lines did the core deliver. Both fields are captured together.



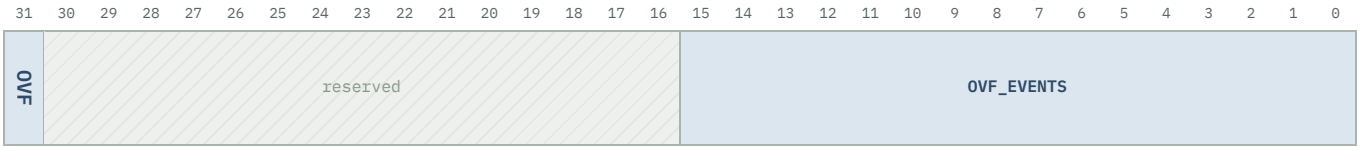
BITS	FIELD	ACCESS	RESET	DESCRIPTION
[31:16]	FRAMES	RO	0	Frame Starts seen by the output stage since reset (wraps).
[15:0]	LINES	RO	0	Lines delivered in the last complete frame, after the crop.

© 2026 ApotheoTech LLC | apotheotech.com | info@apotheotech.com

0x44 VID_OVF_CNT RO

reset 0x00000000

Records frames cut or dropped because the output FIFO filled: a NATIVE line longer than the line buffer, or an AXI4-Stream sink that held tready low too long.



BITS	FIELD	ACCESS	RESET	DESCRIPTION
[31]	OVF	RO	0	Sticky overflow flag. Cleared by reset or VID_CTRL.OVF_CLR.
[30:16]	reserved		0	Reserved, reads 0.
[15:0]	OVF_EVENTS	RO	0	Frames cut or dropped by an overflow. Stops at 0xFFFF.

4.7 Embedded data output

Present when the embedded data output is built (`HWCFG.EBD_OUT` = 1). Without it both read 0 and the two `VID_CTRL` bits have no effect, so software that finds `HWCFG.EBD_OUT` = 0 should not show them.

0x58 EBD_FRAME_INFO RO

reset 0x00000000

Counts the frames that sent embedded data lines and the lines of the latest such frame. Reads 0 when the embedded data output is not built (HWCFG.EBD_OUT = 0).



BITS	FIELD	ACCESS	RESET	DESCRIPTION
[31:16]	FRAMES	RO	0	Frames that sent at least one embedded data line (wraps).
[15:0]	LINES	RO	0	Embedded data lines of the latest frame that sent any; counts up while its lines go out.

© 2026 ApotheoTech LLC | apotheotech.com | info@apotheotech.com

0x5C EBD_OVF_CNT RO

reset 0x00000000

An event is one frame whose embedded data lines were cut or dropped because the sink held m_axis_ebd_tready low until the EBD FIFO filled. The image stream is never affected. A stream reset on m_axis_ebd_aresetn is not an overflow. Reads 0 when the embedded data output is not built.



BITS	FIELD	ACCESS	RESET	DESCRIPTION
[31]	OVF	RO	0	Sticky overflow flag. Cleared by reset or VID_CTRL.EBD_OVF_CLR.
[30:16]	reserved		0	Reserved, reads 0.
[15:0]	OVF_EVENTS	RO	0	Frames whose embedded data lines were cut or dropped. Stops at 0xFFFF.

4.8 Programming ECC_BLOCK

In the ECC build each line payload is split into Basic Blocks of 224 information bytes and a last block with the remainder. `ECC_BLOCK` tells the core where the last block starts. Compute it from the sensor mode:

```
PL = ceil16(Line Length x Pixel Bit / 8)       payload length in bytes, rounded up to 16
n  = floor(PL / 224)
r  = PL mod 224

r != 0 :  ECC_BLOCK = (r   << 16) | n          n Basic Blocks, then a last block of r bytes
r == 0 :  ECC_BLOCK = (224 << 16) | (n - 1)    the last Basic Block is the last block
```

Example: 5328 pixels at RAW12 is 7,992 bytes, rounded up to PL = 8,000. Then n = 35 and r = 160, so `ECC_BLOCK` = 0x00A00023. When the sensor reports the line length in the packet header, the core derives the same layout itself; keep `ECC_BLOCK` programmed for the line length the sensor sends.

Table 4-2. ECC_BLOCK values for common line formats.

MODE	LINE LENGTH	PIXEL BIT	PL (BYTES)	N	LAST BLOCK	ECC_BLOCK
IMX530 full width	5328	RAW12	8000	35	160 B	0x00A00023
IMX547 full width	2472	RAW12	3712	16	128 B	0x00800010
4K UHD	3840	RAW8	3840	17	32 B	0x00200011
Full HD	1920	RAW10	2400	10	160 B	0x00A0000A
Exact blocks	1792	RAW8	1792	8	224 B (last Basic Block)	0x00E00007

© 2026 ApotheoTech LLC | apotheotech.com | info@apotheotech.com

4.9 Programming sequence

01 Hold the link in reset

Keep `rstn_i` low until the PHY reports the lanes ready.

02 Identify the core

Read `ID` (expect 0xA715EC01) and `VERSION` (0x03040001 for v3.4.1). Read `HWCFG` for the lane count, build, output and embedded data output.

03 Configure the link

Write `PIX_BIT` and `LINE_LENGTH`. CRC build: write `CRC_OPTION` = 1 and set the sensor's CRC Option on. ECC build: write `ECC_OPTION` (1 or 2, as the sensor) and `ECC_BLOCK`.

04 Optional crop

Write `VID_LINE_SKIP`, `VID_WIDTH` and `VID_HEIGHT`. These may also be changed later, while streaming.

05 Start the sensor

Release `rstn_i` and start streaming from the sensor.

06 Check the link

Read `VID_FRAME_INFO`: `FRAMES` should count up with every frame and `LINES` should equal the image height after the crop. `HDR_CRC_ERR_CNT` and `VID_OVF_CNT` should stay 0 and `CRC_ERR_LINE[31]` should read 0. With the embedded data output built, `EBD_FRAME_INFO` counts the frames that sent metadata.

07 Monitor

Read the error counters periodically: `HDR_CRC_ERR_CNT` in both builds, `PAY_CRC_ERR_CNT` and `CRC_ERR_LINE` in the CRC build, `C_ERR_CNT` and `ECC_FAIL_CNT` in the ECC build. Clear them with `ERR_CNT_CTRL.CLEAR`, and the overflow status with `VID_CTRL.OVF_CLR` and `VID_CTRL.EBD_OVF_CLR`.

The same sequence with plain register accesses, for a 2-lane CRC build receiving 5328-pixel RAW12 lines with the CRC footer, cropped to 1920 x 1080:

© 2026 ApotheoTech LLC | apotheotech.com | info@apotheotech.com

```
/* base = address of slvs_ec_rx_0 in the Vivado address editor */
#define RX(off)  (base + (off))

if (Xil_In32(RX(0x00)) != 0xA715EC01) return -1;        /* ID              */
u32 ver   = Xil_In32(RX(0x04));                          /* VERSION 3.4.1   */
u32 hwcfg = Xil_In32(RX(0x08));                          /* lanes, build    */

Xil_Out32(RX(0x10), 12);       /* PIX_BIT       RAW12              */
Xil_Out32(RX(0x14), 5328);     /* LINE_LENGTH   pixels             */
Xil_Out32(RX(0x1C), 1);        /* CRC_OPTION    check CRC32 footer */
Xil_Out32(RX(0x30), 8);        /* VID_LINE_SKIP                    */
Xil_Out32(RX(0x34), 1920);     /* VID_WIDTH                        */
Xil_Out32(RX(0x38), 1080);     /* VID_HEIGHT                       */

/* ... release rstn_i, start the sensor ... */

u32 info   = Xil_In32(RX(0x40));   /* VID_FRAME_INFO                    */
u32 frames = info >> 16;             /* counts up every frame           */
u32 lines  = info & 0xFFFF;          /* 1080 after the crop             */
if (Xil_In32(RX(0x54)) & 0x80000000) { /* CRC_ERR_LINE: an error since clear */
    u32 hdr = Xil_In32(RX(0x4C));        /* bad copies [15:0], lost [31:16]     */
    u32 pay = Xil_In32(RX(0x50));        /* payload CRC32 failures              */
}
```

The software driver supplied with the core performs these steps for you, including the `ECC_BLOCK` computation; see Software.

© 2026 ApotheoTech LLC | apotheotech.com | info@apotheotech.com

SECTION 05

Example Designs and Software

Evaluation platforms, ready-to-build example designs, the bring-up flow and the software that ship with the core.

5.1 Evaluation platforms

Example designs target three boards. Each connects an SLVS-EC sensor module through an FMC adapter and runs the core with either bundled PHY.

Table 5-1. Evaluation platforms.

BOARD	DEVICE	PHY OPTIONS	EXAMPLE DESIGNS
AMD ZCU106	Zynq UltraScale+ MPSoC XCZU7EV	GT Wizard (GTH), HP I/O	Basic streaming, DisplayPort output, demo bitstream
ApotheoTech Aper-Oculus	Kria K26 SOM, Zynq UltraScale+ MPSoC	GT Wizard (GTH), HP I/O	Basic streaming, DisplayPort output
AMD KCU116	Kintex UltraScale+ XCKU5P	GT Wizard (GTY), HP I/O	Basic streaming
Your board	Any supported AMD device	GT Wizard or HP I/O	Custom integration (NRE)

5.2 Demo bitstreams

Pre-built bitstreams let you see your sensor streaming before you build anything.

Table 5-2. Available demo configurations. Demo bitstreams are supplied as .bit files; ask for other platforms.

BOARD	SENSOR	VIDEO OUTPUT	DESCRIPTION
ZCU106	IMX530	DisplayPort	Full demo with a live 4K30 display and on-screen link status

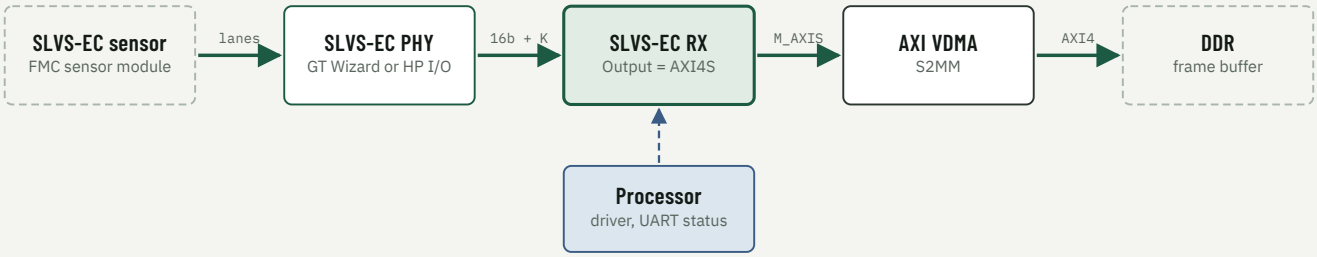
5.3 Example designs

Every example design is a complete, synthesizable Vivado project: the example top level in RTL source, constraints, a script that recreates the project, the bundled PHY and a bare-metal application for the processor.

© 2026 ApotheoTech LLC | apotheotech.com | info@apotheotech.com

Basic streaming

Included

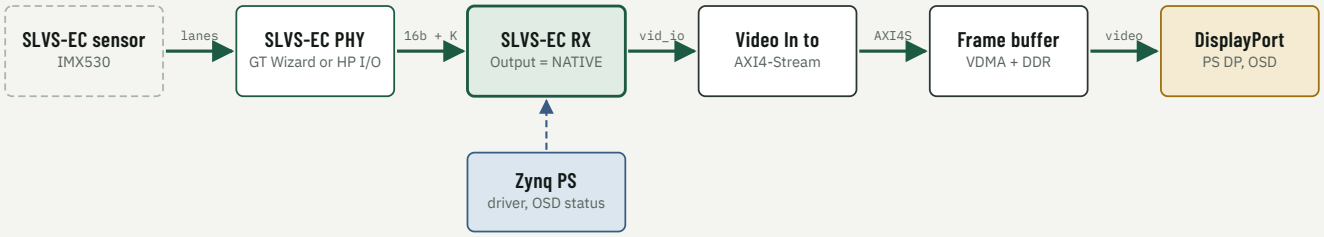


The receiver with Output = AXI4S writes frames through an AXI VDMA into a DDR frame buffer. The processor configures the sensor and the core and reports link status, frame count and error counters over the UART. The starting point for most custom designs.

Output AXI4S **Platforms** ZCU106, Aper-Oculus, KCU116 **Software** bare-metal, UART console

DisplayPort output

Included with Zynq MPSoC targets



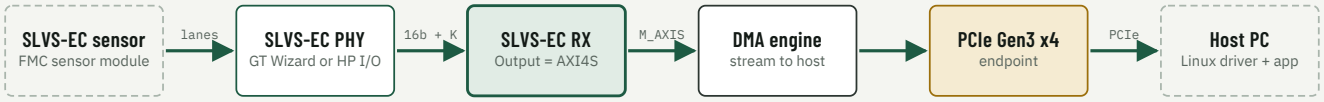
Live video on a DisplayPort monitor with an on-screen status overlay. The receiver's NATIVE output feeds AMD Video In to AXI4-Stream, a frame buffer and the processing system's DisplayPort controller. Zynq UltraScale+ MPSoC targets only (ZCU106 and Aper-Oculus); not available for KCU116.

Output NATIVE **Platforms** ZCU106, Aper-Oculus **Software** bare-metal, DisplayPort and OSD

© 2026 ApotheoTech LLC | apotheotech.com | info@apotheotech.com

PCIe DMA

Add-on



High-performance PCIe Gen3 x4 DMA that streams frames into host PC memory. Includes a boilerplate Linux character device driver and a user-space capture application.

Output AXI4S **Host** Linux, character device **Link** PCIe Gen3 x4

Custom integration

NRE

Work directly with ApotheoTech on a design for your sensor, board and pipeline: a new sensor mode, a different FPGA family, a CoaXPress, SDI or Ethernet output, or a complete camera.

5.4 Bring-up flow

01 Recreate the project

Run the example design's Tcl script in Vivado. It adds the IP repository, builds the block design and applies the constraints.

02 Choose the build

In the IP GUI set Lanes and Payload protection to match the sensor mode, pick the Output, and size the Output FIFO for the longest line (NATIVE) or the longest sink stall (AXI4S). Turn the Embedded data output on if the application reads the sensor's metadata. Use the GT Wizard PHY for Baud Grade 3, or either PHY for Baud Grade 1 and 2.

03 Build and export

Generate the bitstream and export the hardware platform for Vitis.

04 Build the application

Create the Vitis platform and application, add the example sources and the driver, and select the sensor mode.

05 Run

Program the board. The console prints the IP version and build, then starts the sensor.

© 2026 ApotheoTech LLC | apotheotech.com | info@apotheotech.com

06 Check the link

The frame counter rises with every frame, the delivered height matches the image and the header CRC error count stays at 0. A rising header error count points at lane order, polarity or signal integrity.

5.5 Software

The core comes with a software driver for setup and control, supplied on purchase, for MicroBlaze, Zynq-7000 and Zynq UltraScale+ MPSoC processors. It identifies the core and its build, programs the link for a sensor mode, checks that the payload protection matches the build, sets the crop, controls the embedded data output, and reads the frame, overflow and error counters. The example designs use it, so they double as working references for your own application. The PCIe DMA add-on adds a Linux character device driver and a user-space application.

5.6 Verification

Release 3.4.1 is verified on the AXI4S and NATIVE outputs with two independent benches. A VHDL testbench with its own SLVS-EC transmitter model runs a matrix over every lane count, pixel depth and payload protection setting, with embedded data lines at the start and the end of the frame, and checks every image line and every embedded data line on both outputs. A UVM 1.2 testbench drives the lanes from a transmitter model and AXI4-Lite from a register model, and a reference model checks every line and every status change, with protocol monitors on the AXI4-Stream rules on every clock.

4,846	264	0
matrix runs passing on AXI4S and NATIVE: 1 to 8 lanes, RAW8 to RAW16, CRC and ECC, with embedded data	UVM regression runs passing: 22 tests, 3 seeds, 4 builds	mismatches against the Reed-Solomon reference decoder over 30,000 vectors

- Line Length sweeps at 4, 6 and 8 lanes cover every case where a line ends on a partial beat, in both builds.
- Errors are injected and accounted for: corrupted header copies, payload CRC32 errors, symbol errors in every Reed-Solomon block, on image and embedded data lines alike. Every error is corrected or counted, and the CRC error registers are read back at the end of every run.
- ECC results are checked bit for bit against a C reference decoder, for both ECC Options, up to two errors beyond the correction limit per block.
- The output stages are run with stream clocks from 1.1 to 12 ns, sink stalls, overflow and recovery, and stream resets, on the image and the embedded data stream.
- The AXI4-Lite interface is stress-tested with random backpressure and multiple outstanding requests.
- The receiver has run in hardware with the Sony IMX530 on Zynq UltraScale+ MPSoC.

© 2026 ApotheoTech LLC | apotheotech.com | info@apotheotech.com

SECTION 06

Resources and Ordering

Device and PHY support, resource use, what you receive with a license, and how to evaluate and order the core.

6.1 Supported devices

The core ships with two PHYs. The APT-SLVS-EC GT Wizard runs on the device's multi-gigabit transceivers (MGT) up to Baud Grade 3. The SLVS-EC HP I/O PHY runs on high-performance I/O banks up to Baud Grade 2 and needs no transceivers.

Table 6-1. AMD device and PHY support.

FAMILY	GT WIZARD PHY (MGT) Baud Grade 1 to 3	HP I/O PHY Baud Grade 1 and 2	STATUS
Zynq UltraScale+ MPSoC, including Kria K26	GTH	HP I/O	Pre-validated
Kintex UltraScale+	GTH, GTY	HP I/O	Pre-validated
Artix UltraScale+	GTH	HP I/O	Pre-validated
Other AMD families	GTH, GTY	HP I/O	Contact ApotheoTech
Intel (Altera) FPGAs	Per family		NRE engagement

6.2 Resource utilization

Table 6-2. Typical resource use at Baud Grade 3.

CONFIGURATION	LINE RATE	LUTS	REGISTERS	BRAM
1 lane, Baud Grade 3	5 Gb/s	~4,800	~4,200	6
2 lanes, Baud Grade 3	10 Gb/s	~5,600	~5,400	10
4 lanes, Baud Grade 3	20 Gb/s	~6,600	~6,400	18
8 lanes, Baud Grade 3	40 Gb/s	~7,600	~7,400	34

Excludes the PHY and user logic. Results from Vivado 2025.2; subject to change with updates. Contact ApotheoTech for figures for a specific build.

The output FIFO is the main memory user and scales with the depth you pick. The estimate below helps budget block RAM. The embedded data FIFO, when built, has the same word width and adds memory at the same rate for its own depth: 1,024 words on 4 lanes is about 4 BRAM36.

Table 6-3. Output FIFO memory, estimated BRAM36 blocks by depth and lane count. At 128 and 256 words Vivado may build the FIFO from LUTs.

DEPTH	1 LANE	2 LANES	4 LANES	6 LANES	8 LANES

© 2026 ApotheoTech LLC | apotheotech.com | info@apotheotech.com

DEPTH	1 LANE	2 LANES	4 LANES	6 LANES	8 LANES
128	1	1	2	3	4
256	1	1	2	3	4
512	1	1	2	3	4
1,024	1	2	4	6	8
2,048 (default)	2	4	8	11	15
4,096	4	8	15	22	29
8,192	9	17	33	49	65
16,384	18	34	66	98	130

6.3 Deliverables

Table 6-4. What a license includes.

IP core	Encrypted netlist for the target FPGA family, packaged for Vivado IP Integrator with its customization GUI
PHY	APT-SLVS-EC GT Wizard (GTH and GTY transceivers, Baud Grade 1 to 3) and SLVS-EC HP I/O PHY (Baud Grade 1 and 2), set up for your sensor mode
Integration guide	Complete signal descriptions, register map and integration notes
Instantiation templates	Verilog and VHDL wrapper templates
Example designs	Complete Vivado projects with RTL source, constraints and the bundled PHY
Software	Driver for setup and control on MicroBlaze, Zynq-7000 and Zynq UltraScale+ MPSoC, plus the example applications
Technical support	Email support included with the license; premium support available

6.4 Evaluation and ordering

Evaluate

Demo bitstreams for supported boards, so you can see your sensor streaming before you commit. Ask for a platform not listed.

License

Perpetual license with annual maintenance for updates and support. Add-on example designs and NRE engineering on request.

© 2026 ApotheoTech LLC | apotheotech.com | info@apotheotech.com

Engineering services

Custom integration on your board, new sensor modes, and complete camera designs from sensor to CoaXPress, PCIe or Ethernet.

CONTACT

ApotheoTech LLC

Email: info@apotheotech.com

Web: apotheotech.com

For pricing, evaluation access or technical questions, email us with your sensor, lane count and target device.

Related ApotheoTech products: CoaXPress 2.1 Host and Device IP cores, an NVMe host IP core, and the Aper-Oculus Kria K26 carrier board.

6.5 Revision history

Table 6-5. Datasheet revisions.

REVISION	DATE	CHANGES
2.1	November 2026	Updated for IP v3.4.1. Embedded data lines now leave the core on a separate AXI4-Stream, `M_AXIS_EBD`, with its own clock and FIFO and available with either video output; this replaces the in-band option described in Rev 2.0. CRC error registers at 0x4C to 0x54 as built: bad header copies and lost headers, payload CRC32 count, and the Line Number of the last failing line with a sticky error flag. Embedded data registers at 0x58 and 0x5C, and the matching `HWCFG` and `VID_CTRL` fields (24 registers). Line Length rule per lane count, partial last beat, payload protection rule. Verification summary for 3.4.1.
2.0	November 2026	Rewritten for IP v3.3: 6-lane builds, RAW14, the Reed-Solomon ECC build, the AXI4S and NATIVE outputs, optional embedded data lines, header and payload CRC error counters with last-line status, crop and overflow handling, bundled GT Wizard and HP I/O PHYs, full register map, interfaces and timing captures, example designs and verification summary. Line rates stated per Baud Grade. Sensor table corrected to Sony product information.
1.1	September 2026	Previous release.

6.6 Notices

SLVS-EC is a trademark of Sony Group Corporation or its affiliates. Pregius and Pregius S are trademarks of Sony Group Corporation or its affiliates. AMD, Xilinx, Zynq, Kria, Kintex, Artix, UltraScale, Vivado and Vitis are trademarks of Advanced Micro Devices, Inc. Intel and Altera are trademarks of Intel Corporation or its subsidiaries. All other names are the property of their respective owners. JIIA EVI-001-2019 is published by the Japan Industrial Imaging Association. Specifications are subject to change without notice. This document describes the IP core and does not grant any license.