UG0806
User Guide
MIPI CSI-2 Receiver Decoder For PolarFire

Microsemi
a Microchip company

Microsemi Headquarters
One Enterprise, Aliso Viejo,
CA 92656 USA
Within the USA: +1 (800) 713-4113
Outside the USA: +1 (949) 380-6100
Sales: +1 (949) 380-6136
Fax: +1 (949) 215-4996
Email: sales.support@microsemi.com
www.microsemi.com

About Microsemi

Microsemi, a wholly owned subsidiary of Microchip Technology Inc. (Nasdaq: MCHP), offers a comprehensive portfolio of semiconductor and system solutions for aerospace & defense, communications, data center and industrial markets. Products include high-performance and radiation-hardened analog mixed-signal integrated circuits, FPGAs, SoCs and ASICs; power management products; timing and synchronization devices and precise time solutions, setting the world's standard for time; voice processing devices; RF solutions; discrete components; enterprise storage and communication solutions, security technologies and scalable anti-tamper products; Ethernet solutions; Power-over-Ethernet ICs and midspans; as well as custom design capabilities and services. Learn more at www.microsemi.com.

Contents

Figures

Tables

1 Revision History

The revision history describes the changes that were implemented in the document. The changes are listed by revision, starting with the current publication.

1.1 Revision 4.0

Updated the document for Libero SoC v12.1.

1.2 Revision 3.0

The following is a summary of changes made in this revision.

- Support for RAW12 data type was added.
- Added frame_valid_o output signal in the IP, see Table 2, page 5.
- Added g_NUM_OF_PIXELS configuration parameter in Table 3, page 6.

1.3 Revision 2.0

Support for RAW10 data type was added.

1.4 Revision 1.0

The first publication of this document.

2 Introduction

MIPI CSI-2 is a standard specification defined by mobile industry processor interface (MIPI) alliance. The camera serial interface 2 (CSI-2) specification defines an interface between a peripheral device (camera) and a host processor (base-band, application engine). This user guide describes the MIPI CSI-2 receiver decoder for PolarFire (MIPI CSI-2 RxDecoder), which decodes the data from the sensor interface. The IP core supports multi-lane (1,2 and 4 lanes) for RAW8, RAW10, and RAW12 data types. MIPI CSI-2 operates in two modes—high-speed mode and low-power mode. In high-speed mode, MIPI CSI-2 supports the transport of image data using short packet and long packet formats. Short packets provide frame synchronization and line synchronization information. Long packets provide the pixel information. The sequence of transmitted packets is as follows.

1. Frame start (short packet)
2. Line start (optional)
3. Few image data packets (long packets)
4. Line end (optional)
5. Frame end (short packet)

One long packet is equivalent to one line of image data. The following illustration shows the video data stream.

Figure 1 • **Video Data Stream**

3 Hardware Implementation

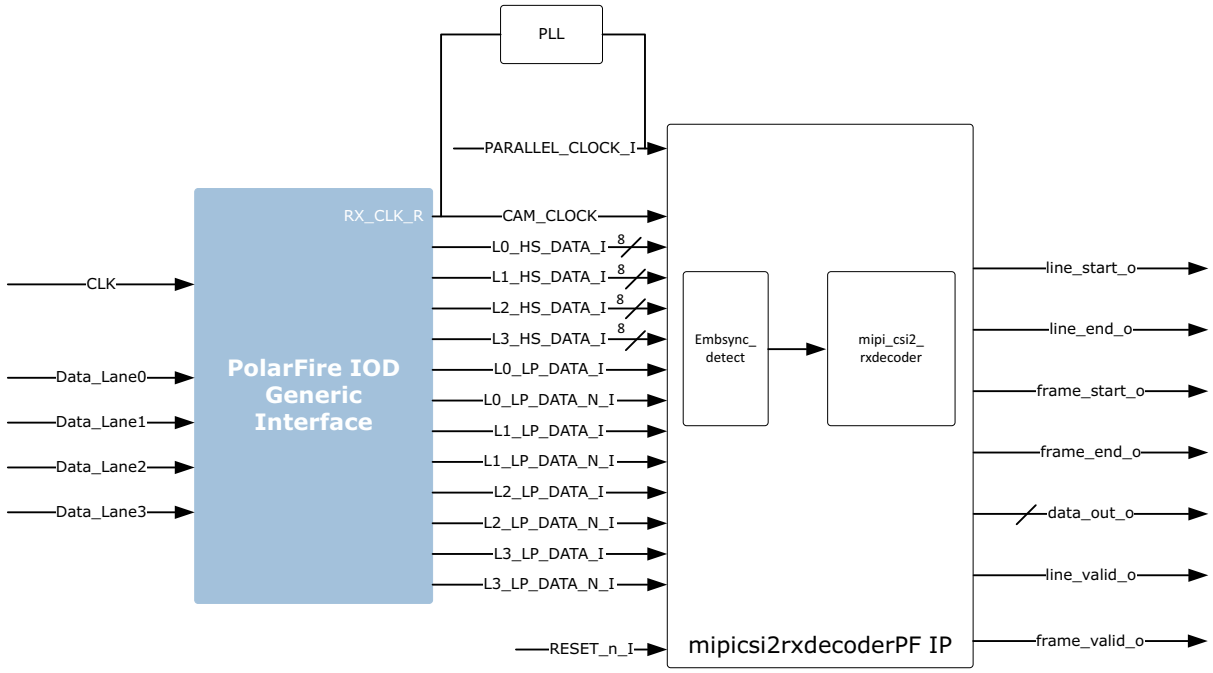
This section describes the hardware implementation details. The following illustration shows the MIPI CSI2 receiver solution that contains the MIPI CSI2 RxDecoder IP. This IP has to be used in conjunction with the PolarFire® MIPI IOD generic interface blocks and phase-locked loop (PLL). The MIPI CSI2 RxDecoder IP is designed to work with the PolarFIre MIPI IOG blocks. Figure 2, page 3 shows the pin connection from the PolarFire IOG to the MIPI CSI2 RxDecoder IP. A PLL is required to generate the parallel clock (pixel clock). The input clock to the PLL will be from the RX_CLK_R output pin of the IOG. The PLL has to be configured to produce the parallel clock, based on the MIPI_bit_clk and the number of lanes used. The equation used to calculate the parallel clock is as follows.

$$CAM\_CLOCK\_I = (MIPI\_bit\_clk)/4$$

$$PARALLEL\_CLOCK = (CAM\_CLOCK\_I \times Num\_of\_Lanes \times 8)/(g\_DATAWIDTH \times g\_NUM\_OF\_PIXELS)$$

The following illustration shows the architecture of MIPI CSI-2 Rx for PolarFire.

Figure 2 • **Architecture of MIPI CSI-2 Rx Solution**



The preceding figure shows the different modules in the MIPI CSI2 RxDecoder IP. When used in conjunction with the PolarFire IOD Generic and PLL, this IP can receive and decode the MIPI CSI2 packets to produce pixel data along with the valid signals.

3.1 Design Description

This section describes the different internal modules of the IP.

3.1.1 Embsync_detect

This module receives data from the PolarFire IOG and detects the embedded SYNC code in the received data of each lane. This module also aligns the data from each lane to the SYNC code and sends it to the mipi_csi2_rxdecoder module for decoding the packet.

3.1.2 mipi_csi2_rxdecoder

This module decodes the incoming short packets and long packets and generates the `frame_start_o, frame_end_o, frame_valid_o, line_start_o, line_end_o, word_count_o, line_valid_o, and data_out_o` outputs. Pixel data arrives between line start and line end signals. Short packet contains only packet header and supports various data types. MIPI CSI-2 Receiver IP Core supports the following data types for short packets.

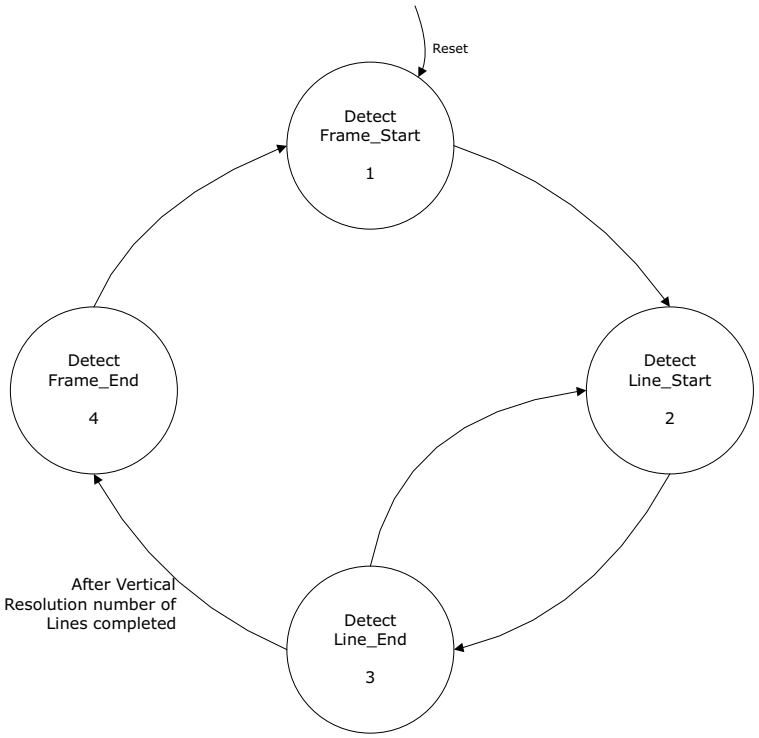
Table 1 • **Supported Data Types**

Data Type	Description
0x00	Frame Start
0x01	Frame End

Long packet contains the image data. The length of the packet is determined by the horizontal resolution, to which the camera sensor is configured. This can be seen at the word_count_o output signal.

The following illustration shows the FSM implementation of decoder.

Figure 3 • **FSM Implementation of Decoder**



1. Frame Start: On receiving the frame start packet, generate the frame start pulse and then wait for line start.
2. Line Start: On receiving the line start indication, generate the line start pulse.

3. Line End: On generating the line start pulse, store the pixel data, and then generate the line end pulse. Repeat Step 2 and 3 until the frame end packet is received.
4. Frame End: On receiving the frame end packet, generate the frame end pulse. Repeat the above steps for all frames.

The CAM_CLOCK_I must be configured to the image sensor frequency, to process the incoming data, regardless of Num_of_lanes_i configured to one lane, two lanes, or four lanes.

The IP supports RAW8, RAW10, and RAW12 data types. One pixel per clock is received on data_out_o if the g_NUM_OF_PIXELS is set to one. If g_NUM_OF_PIXELS is set to 4 then four pixels per clock are sent out and the parallel clock has to be configured 4 times lower than the normal case. The four pixel per clock configuration gives user the flexibility to run their design at higher resolutions and higher camera data rate, which makes it easier to meet design timings. To indicate valid image data, line_valid_o output signal is sent. Whenever it is asserted high, output pixel data is valid.

3.2 Inputs and Outputs

The following table lists the input and output ports of the IP configuration parameters.

Table 2 • **Input and Output Ports**

Signal Name	Direction	Width	Description
CAM_CLOCK_I	Input	1	Image sensor clock
PARALLEL_CLOCK_I	Input	1	Pixel clock
RESET_N_I	Input	1	Asynchronous active low reset signal
L0_HS_DATA_I	Input	8-bit	High speed input data from lane 1
L1_HS_DATA_I	Input	8-bit	High speed input data from lane 2
L2_HS_DATA_I	Input	8-bit	High speed input data from lane 3
L3_HS_DATA_I	Input	8-bit	High speed input data from lane 4
L0_LP_DATA_I	Input	1	Positive low power input data from lane one
L0_LP_DATA_N_I	Input	1	Negative low power input data from lane one
L1_LP_DATA_I	Input	1	Positive low power input data from lane two
L1_LP_DATA_N_I	Input	1	Negative low power input data from lane two
L2_LP_DATA_I	Input	1	Positive low power input data from lane three
L2_LP_DATA_N_I	Input	1	Negative low power input data from lane three
L3_LP_DATA_I	Input	1	Positive low power input data from lane four
L3_LP_DATA_N_I	Input	1	Negative low power input data from lane four
data_out_o	Output	g_DATAWIDTH*g_NUM_OF_PIXELS-1 : 0	8-bit,10-bit and 12-bit with single pixel per clock. 32-bit, 40-bit and 48-bit with 4 pixels per clock.
Line_Valid_o	Output	1	Data valid output. Asserted high when `data_out_o` is valid
word_count_o	Output	16-bit	Represents horizontal resolution in hex format
frame_start_o	Output	1	Asserted high for one clock when frame start is detected in the incoming packets
frame_end_o	Output	1	Asserted high for one clock when frame end is detected in the incoming packets
frame_valid_o	Output	1	Asserted high for one clock for all active lines in a frame
line_start_o	Output	1	Asserted high for one clock when line start is detected in the incoming packets

Table 2 • **Input and Output Ports *(continued)***

Signal Name	Direction	Width	Description
line_end_o	Output	1	Asserted high for one clock when line end is detected in the incoming packets

3.3 Configuration Parameters

The following table lists the description of the configuration parameters used in the hardware implementation of MIPI CSI-2 Rx Decoder block. They are generic parameters and can vary based on the application requirements.

Table 3 • **Configuration Parameters**

Name	Description
g_DATAWIDTH	Input pixel data width. Supports 8-bit, 10-bit, and 12-bit
g_LANE_WIDTH	Number of MIPI lanes. Supports 1, 2, 4 lanes
g_NUM_OF_PIXELS	The following options are available: 1: One pixel per clock 4: Four pixels per clock with pixel clock frequency reduced four times
g_INPUT_DATA_INVERT	The options to invert the incoming data are as follows. 0: does not invert the incoming data 1: inverts the incoming data
g_BUFF_DEPTH	Depth of the line buffer, which is equal to active horizontal resolution.

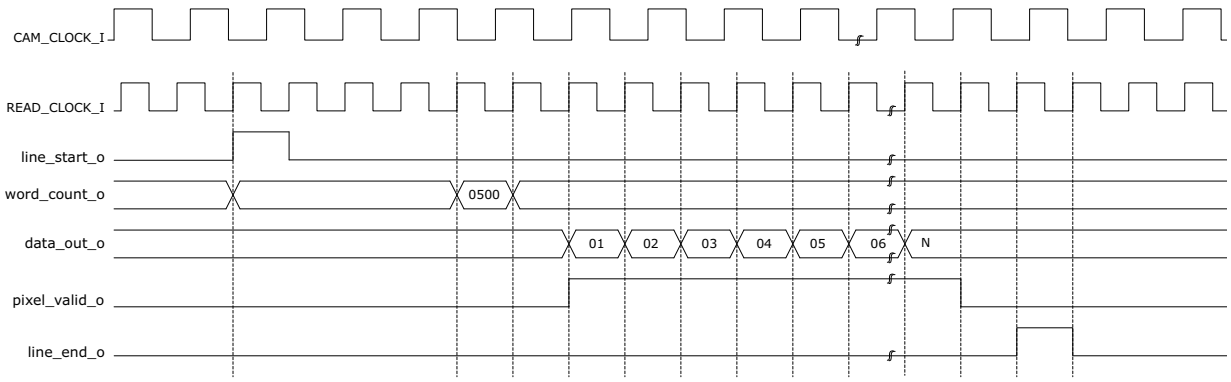
3.4 Timing Diagram

The following sections show the timing diagrams.

3.4.1 Long Packet

The following illustration shows the timing waveform of the long packet.

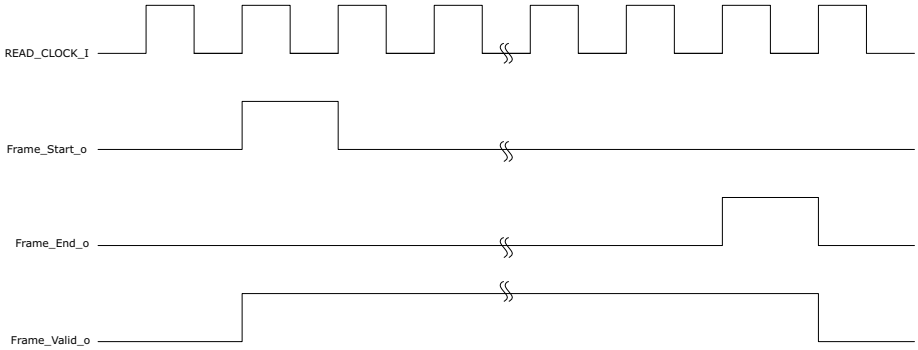
Figure 4 • **Timing Waveform of Long Packet**



3.4.2 Short Packet

The following illustration shows the timing waveform of the frame start packet.

Figure 5 • **Timing Waveform of Frame Start Packet**



3.5 Resource Utilization

The following table shows the resource utilization of a sample MIPI CSI-2 Receiver Core implemented in a PolarFire FPGA (MPF300TS-1FCG1152I package), for RAW8 and 4-lane configuration.

Table 4 • **Resource Utilization**

Element	Usage
DFFs	1263
4-input LUTs	974
LSRAMs	9