# SubLVDS Image Sensor Receiver IP Core - Lattice Radiant Software

# User Guide

## Disclaimers

Lattice makes no warranty, representation, or guarantee regarding the accuracy of information contained in this document or the suitability of its products for any particular purpose. All information herein is provided AS IS and with all faults, and all risk associated with such information is entirely with Buyer. Buyer shall not rely on any data and performance specifications or parameters provided herein. Products sold by Lattice have been subject to limited testing and it is the Buyer's responsibility to independently determine the suitability of any products and to test and verify the same. No Lattice products should be used in conjunction with mission- or safety-critical or any other application in which the failure of Lattice's product could create a situation where personal injury, death, severe property or environmental damage may occur. The information provided in this document is proprietary to Lattice Semiconductor, and Lattice reserves the right to make any changes to the information in this document or to any products at any time without notice.

FPGA-IPUG-02093-1.3

# Contents

# Figures

# Tables

# Acronyms in This Document

A list of acronyms used in this document.

| Acronym | Definition |
| --- | --- |
| LSE | Lattice Synthesis Engine |
| RX | Receive |

# 1.  Introduction

The Lattice Semiconductor SubLVDS Image Sensor Receiver IP Core converts double data rate interface to pixel clock domain. The subLVDS interface is primarily used in image sensors. It has one clock pair and more than one data pairs. The number of data pairs varies, depending on bandwidth requirement.

Compared to LVDS interface, SubLVDS:

- Has lower common mode that is 0.9 V, while the common mode for LVDS is 1.25 V. SubLVDS is typically powered by 1.8 V supply, while LVDS uses 2.5 V supply.
- Has lower differential swing that is ±150 mV, while the differential swing for LVDS is ±175 mV.
- Is a source synchronous interface, the clock pair is running at the same rate as the data. This is not a 7:1 interface.
- Clock is center-aligned with the data.

## 1.1.  Quick Facts

Table 1.1 provides quick facts about the SubLVDS Image Sensor Receiver IP Core.

**Table 1.1. SubLVDS Image Sensor Receiver IP Core Quick Facts**

| IP Requirements | Supported FPGA Family | CrossLink™-NX, Certus™-NX |
|---|---|---|
| **Resource Utilization** | Targeted Devices | LIFCL-40, LIFCL-17, LFD2NX-40 |
| | Supported User Interface | Native interface; see the Signal Description section |
| | Resources | See Table A.1 |
| **Design Tool Support** | Lattice Implementation | IP Core v1.0.x – Lattice Radiant® software 2.1 |
| | Synthesis | Lattice Synthesis Engine |
| | | Synopsys® Synplify Pro® for Lattice |
| | Simulation | For the list of supported simulators, see the Lattice Radiant Software 2.1 User Guide. |

## 1.2.  Features

The key features of the SubLVDS Image Sensor Receiver IP Core include:

- 1 channel
- Supports 4, 6, 8, or 10 data lanes from an image sensor
- Supports 10-bit (RAW10) or 12-bit (RAW12) pixel widths
- Supports gearing of 8 and 16. Gearing 16 option is only for 4-lane configuration
- Can generate XVS and XHS for image sensors operating in Slave mode

The IP Core, however, does not support configuration through registers.

## 1.3. Conventions

### 1.3.1. Nomenclature

The nomenclature used in this document is based on Verilog HDL. This includes radix indications and logical operators. The most significant bit within the pixel data is the highest index.

### 1.3.2. Signal Names

Signal names that end with:

- *_n* are active low
- *_i* are input signals
- *_o* are output signals
- *_io* are bi-directional signals

### 1.3.3. Attribute Names

Attribute names in this document are formatted in title case and italicized (*Attribute Name*).

# 2. Functional Descriptions

## 2.1. Overview

The SubLVDS Image Sensor Receiver IP Core converts double data rate interface into pixel clock domain. The input interface of the design consists of a data bus and a clock in subLVDS interface format. The output interface consists of a 10-bit or 12-bit multi-pixel data, frame valid, line valid, data valid and a pixel clock with a gearing of 1:8 or 1:16.
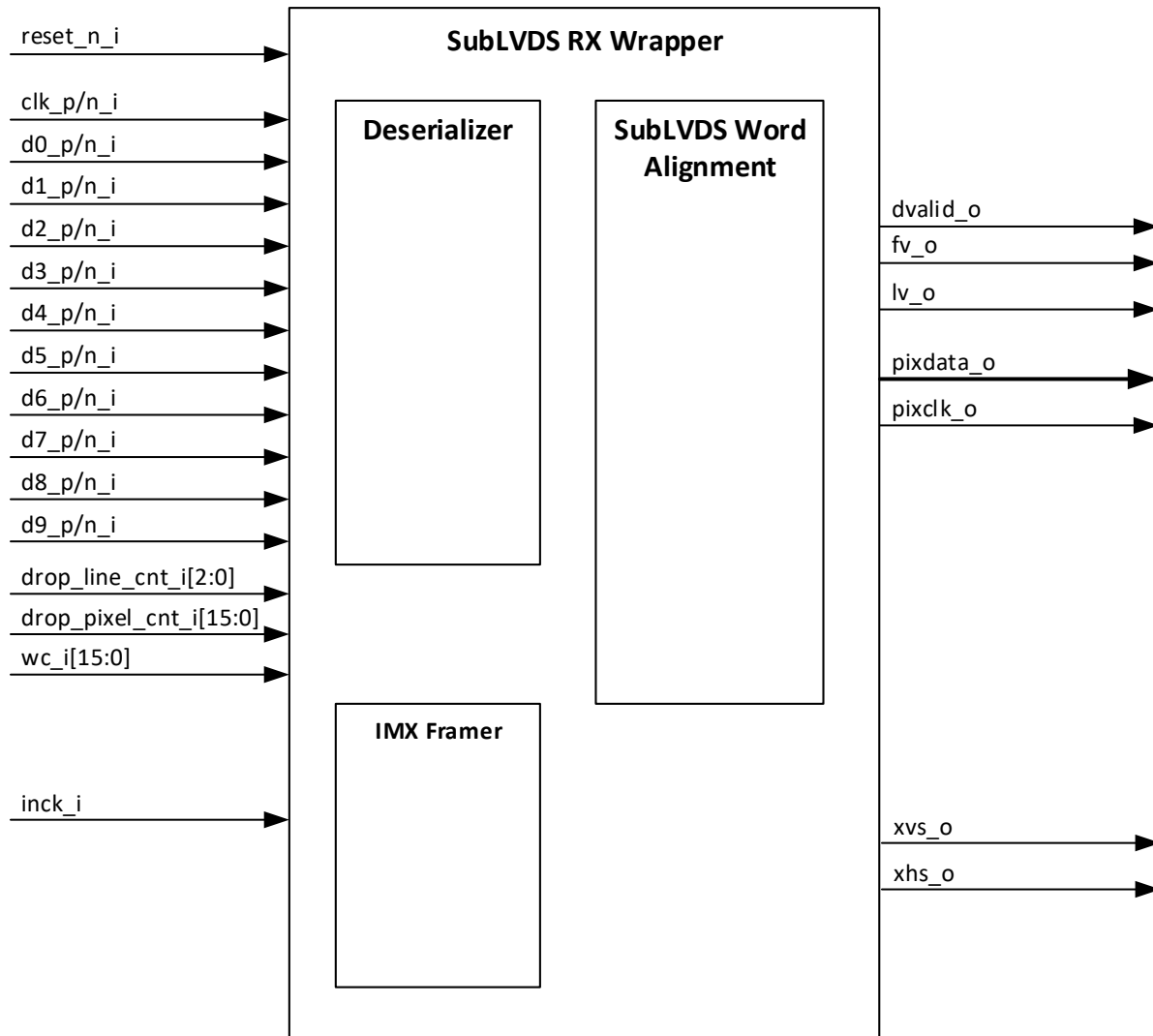
**Figure 2.1. SubLVDS Image Sensor Receiver IP Core Top Level Block Diagram**

## 2.2. Signal Description

Table 2.1 lists top-level input and output signals and their descriptions for the SubLVDS Image Sensor Receiver IP Core.

**Table 2.1. SubLVDS Image Sensor Receiver IP Core Signal Description**

| Port Name | I/O | Width | Description |
|---|---|---|---|
| reset_n_i | In | 1 | System active low asynchronous reset |
| inck_i | In | 1 | IMX Framer input clock. This clock is shared with the Sony Image Sensor. |
| clk_p_i | In | 1 | Positive subLVDS input clock to subLVDS RX |
| clk_n_i | In | 1 | Negative subLVDS input clock to subLVDS RX |
| d0_p_i | In | 1 | Positive subLVDS input data lane 0 to subLVDS RX |
| d0_n_i | In | 1 | Negative subLVDS input data lane 0 to subLVDS RX, complement of d0_p_i |
| d1_p_i | In | 1 | Positive subLVDS input data lane 1 to subLVDS RX |
| d1_n_i | In | 1 | Negative subLVDS input data lane 1 to subLVDS RX, complement of d1_p_i |
| d2_p_i | In | 1 | Positive subLVDS input data lane 2 to subLVDS RX |
| d2_n_i | In | 1 | Negative subLVDS input data lane 2 to subLVDS RX, complement of d2_p_i |
| d3_p_i | In | 1 | Positive subLVDS input data lane 3 to subLVDS RX |
| d3_n_i | In | 1 | Negative subLVDS input data lane 3 to subLVDS RX, complement of d3_p_i |
| d4_p_i | In | 1 | Positive subLVDS input data lane 4 to subLVDS RX |
| d4_n_i | In | 1 | Negative subLVDS input data lane 4 to subLVDS RX, complement of d4_p_i |
| d5_p_i | In | 1 | Positive subLVDS input data lane 5 to subLVDS RX |
| d5_n_i | In | 1 | Negative subLVDS input data lane 5 to subLVDS RX, complement of d5_p_i |
| d6_p_i | In | 1 | Positive subLVDS input data lane 6 to subLVDS RX |
| d6_n_i | In | 1 | Negative subLVDS input data lane 6 to subLVDS RX, complement of d6_p_i |
| d7_p_i | In | 1 | Positive subLVDS input data lane 7 to subLVDS RX |
| d7_n_i | In | 1 | Negative subLVDS input data lane 7 to subLVDS RX, complement of d7_p_i |
| d8_p_i | In | 1 | Positive subLVDS input data lane 8 to subLVDS RX |
| d8_n_i | In | 1 | Negative subLVDS input data lane 8 to subLVDS RX, complement of d8_p_i |
| d9_p_i | In | 1 | Positive subLVDS input data lane 9 to subLVDS RX |
| d9_n_i | In | 1 | Negative subLVDS input data lane 9 to subLVDS RX, complement of d9_p_i |
| drop_line_cnt_i | In | 3 | Number of dropped lines |
| drop_pixel_cnt_i | In | 16 | Number of dropped pixels |
| wc_i | In | 16 | Word count |
| dvalid_o | Out | 1 | Data valid detection signal, indicates valid pixel data |
| fv_o | Out | 1 | Frame valid detection signal, indicates valid frame |
| lv_o | Out | 1 | Line valid detection signal, indicates a valid line |
| pixclk_o | Out | 1 | Pixel clock generated form the CLKDIV |
| pixdata_o | Out | BUS_WIDTH*LANE_WIDTH | Pixel data coming from parser module. Multi-pixel data bus. |
| xvs_o | Out | 1 | Sony slave readout vertical control signal |
| xhs_o | Out | 1 | Sony slave readout horizontal control signal |

**Note:** BUS_WIDTH depends on *Data Type*. When *Data Type* == RAW10, if *RX Gear* == 8, BUS_WIDTH == 10 else BUS_WIDTH == 20.
When *Data Type* == RAW12, if *RX Gear* == 8, BUS_WIDTH == 12 else BUS_WIDTH == 24.
LANE_WIDTH is equal to *Number of RX Lanes*.

## 2.3. Attribute Summary

Table 2.2 lists the parameters used to generate the SubLVDS Image Sensor Receiver IP Core. Table 2.3 describes each attribute.

**Table 2.2. Attributes Table**

| Attribute | Selectable Values | Default | Dependency on Other Attributes |
|---|---|---|---|
| **General** | | | |
| **Receiver** | | | |
| Number of RX Lanes | 4, 6, 8, 10 | 4 | — |
| RX Gear | 8, 16 | 8 | — |
| **Clock** | | | |
| RX Line Rate | 160-1250 | 160 | *Number of RX Lanes, RX Gear* |
| SubLVDS Clock Frequency (MHz) | Calculated | 80 | *RX Line Rate*/2 |
| Pixel Clock Frequency (MHz) | Calculated | 20 | *Pixel Clock Frequency* = 2*(*SubLVDS Clock Frequency/RX Gear*) |
| **Data** | | | |
| Dropped Line Mode | Static, Dynamic | Dynamic | — |
| Dropped Line Count | 0 – 7 | 0 | *Dropped Line Mode == Static* |
| Dropped Pixel Mode | Static, Dynamic | Dynamic | — |
| Dropped Pixel Count | 0 – 65535 | 0 | *Dropped Pixel Mode == Static* |
| Word Count Mode | Static, Dynamic, Off | Dynamic | — |
| Word Count | 0 – 65535 | 0 | *Word Count Mode == Static* |
| **Video** | | | |
| **Video Packet** | | | |
| Data Type | RAW10, RAW12 | RAW10 | — |
| **IMX Framer Settings** | | | |
| Image Sensor Mode | Master, Slave | Master | — |
| V_TOTAL | 2 - 4095 | 10 | *Image Sensor Mode == Slave* |
| H_TOTAL | 0 - 4095 | 1285 | *Image Sensor Mode == Slave* |
| V_H_BLANK | 0 - 4095 | 2 | *Image Sensor Mode == Slave* |

**Table 2.3. Attributes Description**

| Attribute | Description |
|---|---|
| **General** | |
| **Receiver** | |
| Number of RX Lanes | Generates subLVDS I/O |
| RX Gear | Specifies the RX gearing. Only the 4-lane configuration has the option to choose between *RX Gear == 8* or *RX Gear == 16*. |
| **Clock** | |
| RX Line Rate | Target *RX Line Rate* per lane |
| SubLVDS Clock Frequency (MHz) | SubLVDS clock<br>Automatically computed based on target *RX Line Rate*. |
| Pixel Clock Frequency (MHz) | Pixel clock<br>Automatically computed based on target *RX Line Rate*. |
| **Data** | |
| Dropped Line Mode | Allows you to choose between *Static* (predetermined values for the number of dropped lines) and *Dynamic* (user determines values via an added port) Modes. |
| Dropped Line Count | Determines the number of lines to be dropped at the start of the frame. |
| Dropped Pixel Mode | Allows user to choose between *Static* (predetermined values for the number of dropped pixels) and *Dynamic* (user determines values via an additional port) Modes |
| Dropped Pixel Count | Crops the number of pixels after SAV (the OPB and OPB ignore pixels). Refer to the Sony IMX sensor specification for information on these pixels.<br>The input value should be equal to desired number of pixels to drop */ Number of RX Lanes*. For example, to drop 8 pixels when *Number of RX Lanes == 4*, the input to *Dropped Pixel Count* should be 2. |
| Word Count Mode | Allows you to choose between *Static* (predetermined values for the word count), *Dynamic* (user determines values via an additional port) and *Off* (logic not used) Modes. |
| Word Count | Number of active video pixels per line after the dropped pixels (when *Dropped Pixel Count* > 0). Reducing this effectively drops the OPB ignore bits right before the EAV. Kindly refer to the Sony IMX sensor specification for information on these pixels.<br>If *Word Count* == 0 and *Dropped Pixel Count* == 0, the total number of pixels coming out of the design is the total number of active pixels sent by sensor + EAV pixels.<br>The input value should be equal to desired total number of pixels / *Number of RX Lanes*. For example, if the desired total number of pixels is 40 and Number of RX Lane == 4, *Word Count* should be 10. |
| **Video** | |
| **Video Packet** | |
| Data Type | Selects desired data type |
| **IMX Framer Settings** | |
| Image Sensor Mode | Sets the mode of the image sensor. In slave mode, it enables the IMX framer. |
| V_TOTAL | Sets the number of lines XVS is driven high. Only available when *Image Sensor Mode == Slave*. |
| H_TOTAL | Sets the number of INCK clocks XHS is driven high. Only available when *Image Sensor Mode == Slave*. |
| V_H_BLANK | Sets the number of INCK clocks XVS and XHS is driven low. Only available when *Image Sensor Mode == Slave*. |

## 2.4. Modules Description

### 2.4.1. Clock, Reset and Initialization

Active low reset is used in the design with synchronous release. Resets for each clock domain are synchronized to their respective clock domains.

The system reset, reset_n_i, is synchronized to the pixel clock domain and it serves as a reset source for the SubLVDS Word Alignment module.

No special reset sequence is required in this IP.

The RX clock input, clk_p_i, is from an external source (image sensor) and should be connected to a dedicated SubLVDS edge clock pin. The Deserializer block generates a pixel clock, pixclk_o, with a gearing of 1:8 or 1:16 for the pixel data.
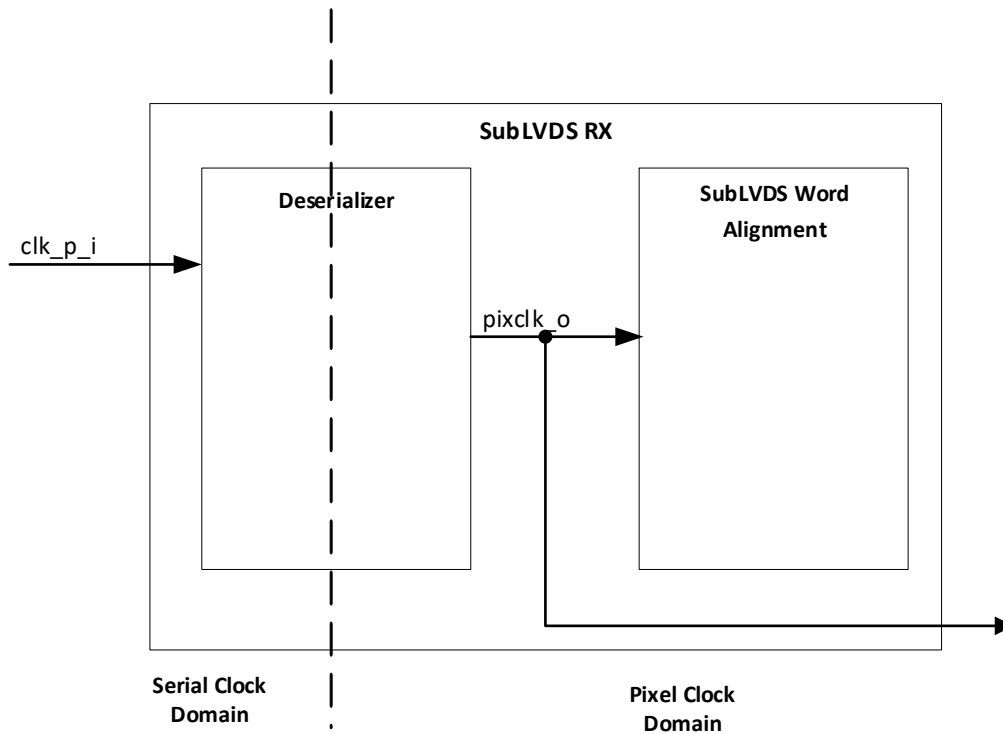


**Figure 2.2. Clock Domain Crossing Block Diagram**

**Table 2.4. Clock Domain Crossing**

| Clock Domain Crossing | Handling Approach |
|---|---|
| SubLVDS Serial Clock to Pixel Clock | 1:8/1:16 gearbox DDR Hard IP |

The general formula for computing the required clocks of the system:

*RX Line Rate* (total) = total pixels (active + blanking) ∗ frame rate ∗ bits per pixel

*RX Line Rate* (per lane) = *RX Line Rate* (total) / *Number of RX lanes*

RX input clock = *RX Line Rate* (per lane)/2

Pixel clock = RX input clock/ *RX Gear*

**Note**: gearing = 4 if 1:8 gearing; 8 if 1:16 gearing

### 2.4.2. SubLVDS Image Sensor Receiver IP Core Submodules

Figure 2.3 shows the detailed block diagram of SubLVDS Image Sensor Receiver IP Core.

The Deserializer block converts each double data rate lane (d*_p_i signals) to a single data rate 8-bit or 16-bit at a slower operating speed within a system.

The word alignment module receives the 8-bit (1:8 gearing) or 16-bit (1:16 gearing) deserialized data (deser_q_o signal) and converts it to 10-bit or 12-bit pixel data according to the set configuration of data type (RAW10 or RAW12). The output of the module is a multi-pixel bus (pixdata_o), pixel clock (pixclk_o), a dvalid_o, fv_o, and lv_o control signals.

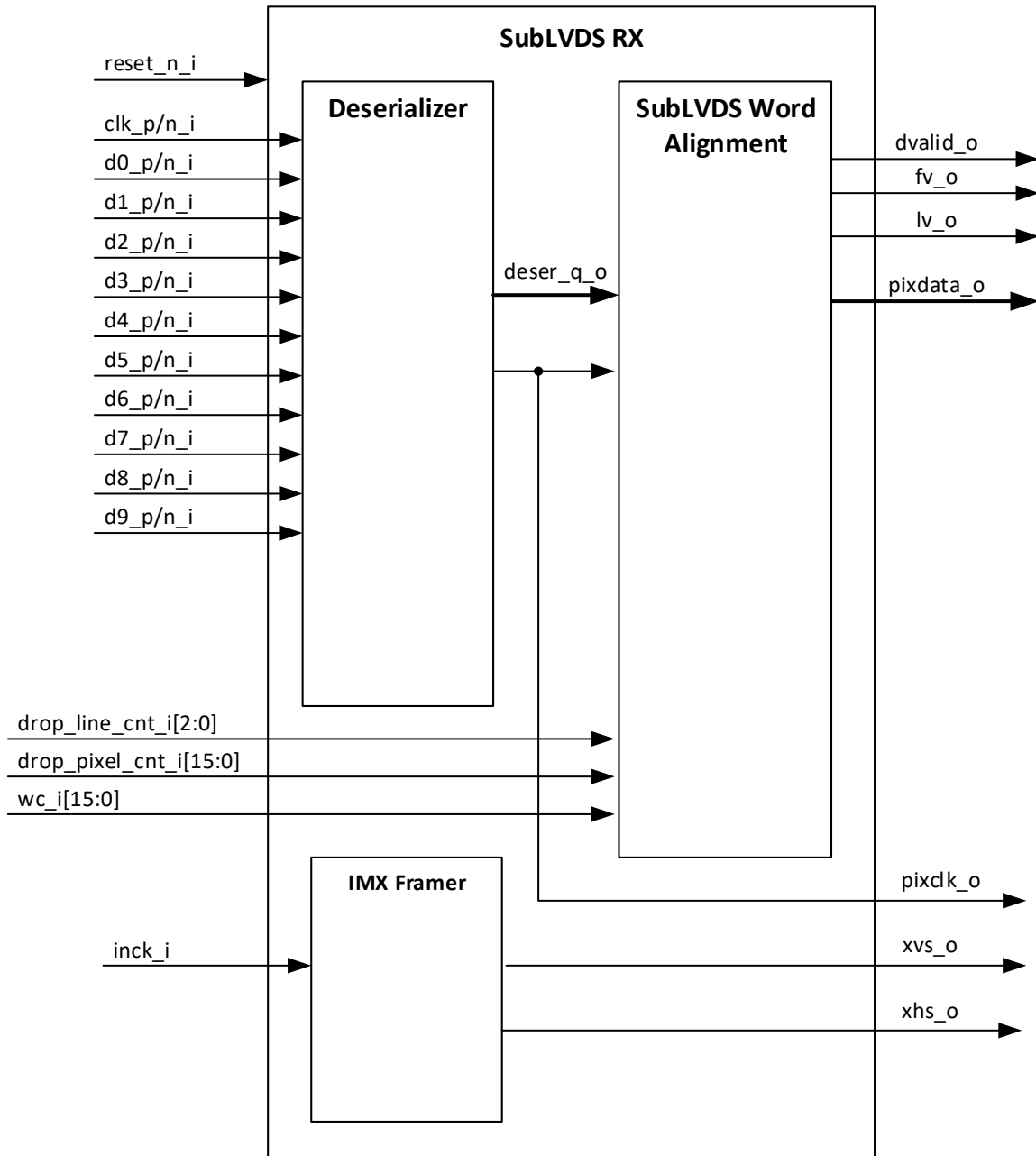The IMX Framer module is used for Image Sensors that operate in the Slave mode only.



**Figure 2.3. SubLVDS Image Sensor Receiver IP Core Detailed Block Diagram**

Figure 2.4 shows the detailed block diagram of Deserializer block when LANE_WIDTH = 4 for the 1:8 gearing case. The number of IDDR components is determined by the parameter LANE_WIDTH.
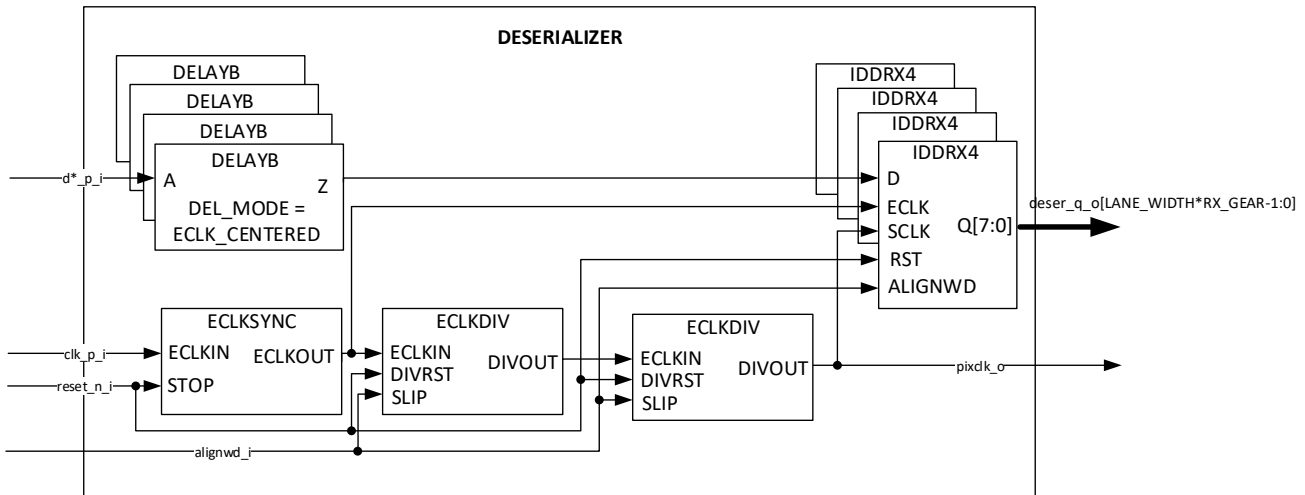


**Figure 2.4. Deserializer of 1:8 Gearing Block Diagram**

Figure 2.5 shows a detailed block diagram of Deserializer block when LANE_WIDTH = 4 for the 1:16 gearing case. FIFO module is used to get to 1:16 gearing.
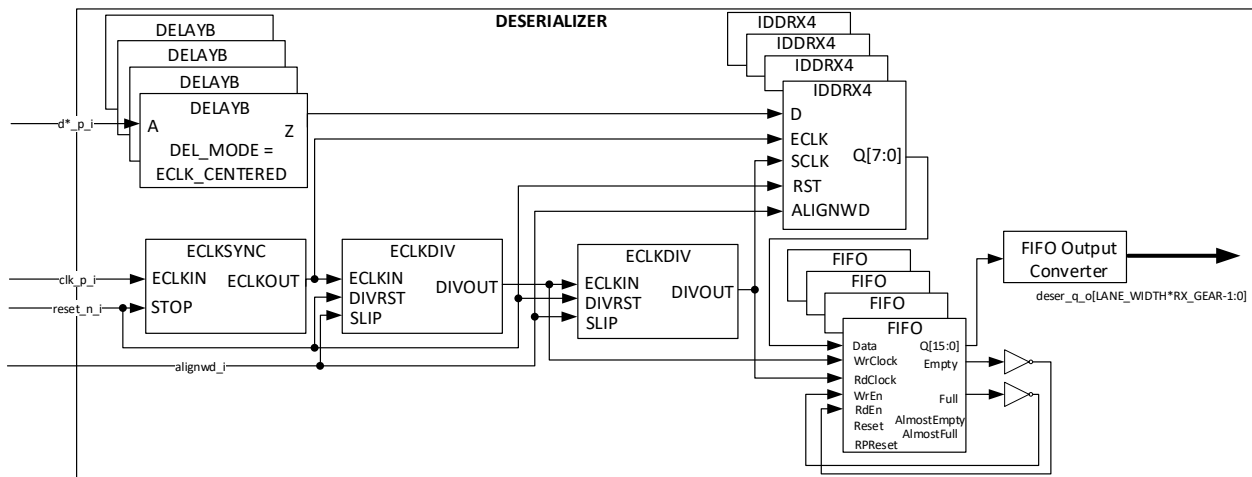


**Figure 2.5. Deserializer of 1:16 Gearing Block Diagram**

To avoid additional clock resource, GDDR_SYNC is not used in this module.

Alignment of clock and data is ensured in the subLVDS word alignment module. Figure 2.6 shows the detailed block diagram of SubLVDS Word Alignment block when LANE_WIDTH = 4. The number of word_aligner instances is determined by the parameter LANE_WIDTH. Sync codes are embedded in each serial data lane by the Sony Image Sensor. The word aligner block detects these sync codes and aligns the deserialized data to a 10-bit or 12-bit pixel data.

The 10-bit or 12-bit data are fed to the parser block. It checks the recognition (sync) codes from the beginning (SAV) and the end (EAV) of each packet if they are part of an active video line or not. The fv_o goes high at the beginning of an active video frame, and low at the end of the frame. Similarly, the lv_o goes active high or low at the beginning or end of an active video line, respectively. The dvalid_o control signal goes active high on clock cycles that have valid pixel data.
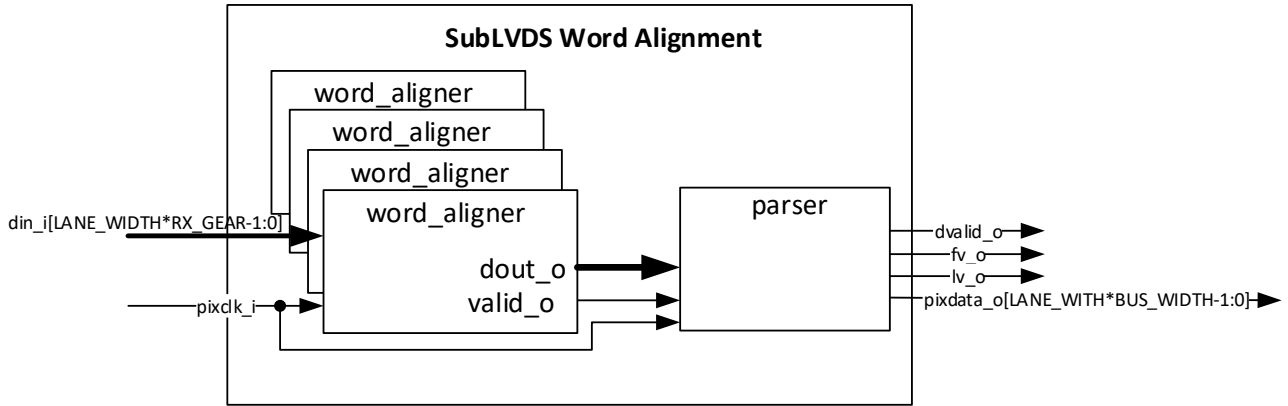
**Figure 2.6. SubLVDS Word Alignment Block Diagram**

**Table 2.5. Indicator States**

| Sync Code | fv State | lv State |
|---|---|---|
| SAV (valid line) | 1 | 1 |
| EAV (valid line) | 1 | 0 |
| SAV (invalid line | 0 | 0 |
| EAV (invalid line) | 0 | 0 |

The IMX Framer module is for Sony Image Sensors that operate in Slave mode. It provides a control mechanism for the rate at which each line and frame is read out. Timing of these two signals is defined in the Sony Image Sensor datasheet.
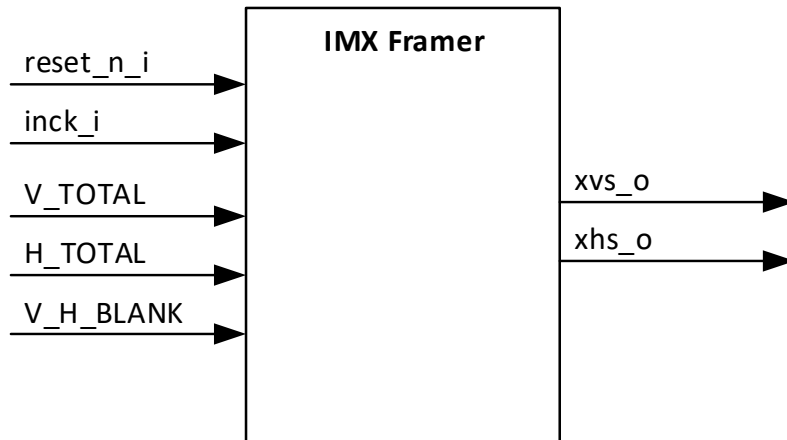


**Figure 2.7. IMX Frame Block Diagram**

## 2.5. Timing Specifications

Figure 2.8 shows the timing of SubLVDS Image Sensor Receiver IP Core input interface. It shows the sync signal and data output timing during 10-bit length serial received from the image sensor.
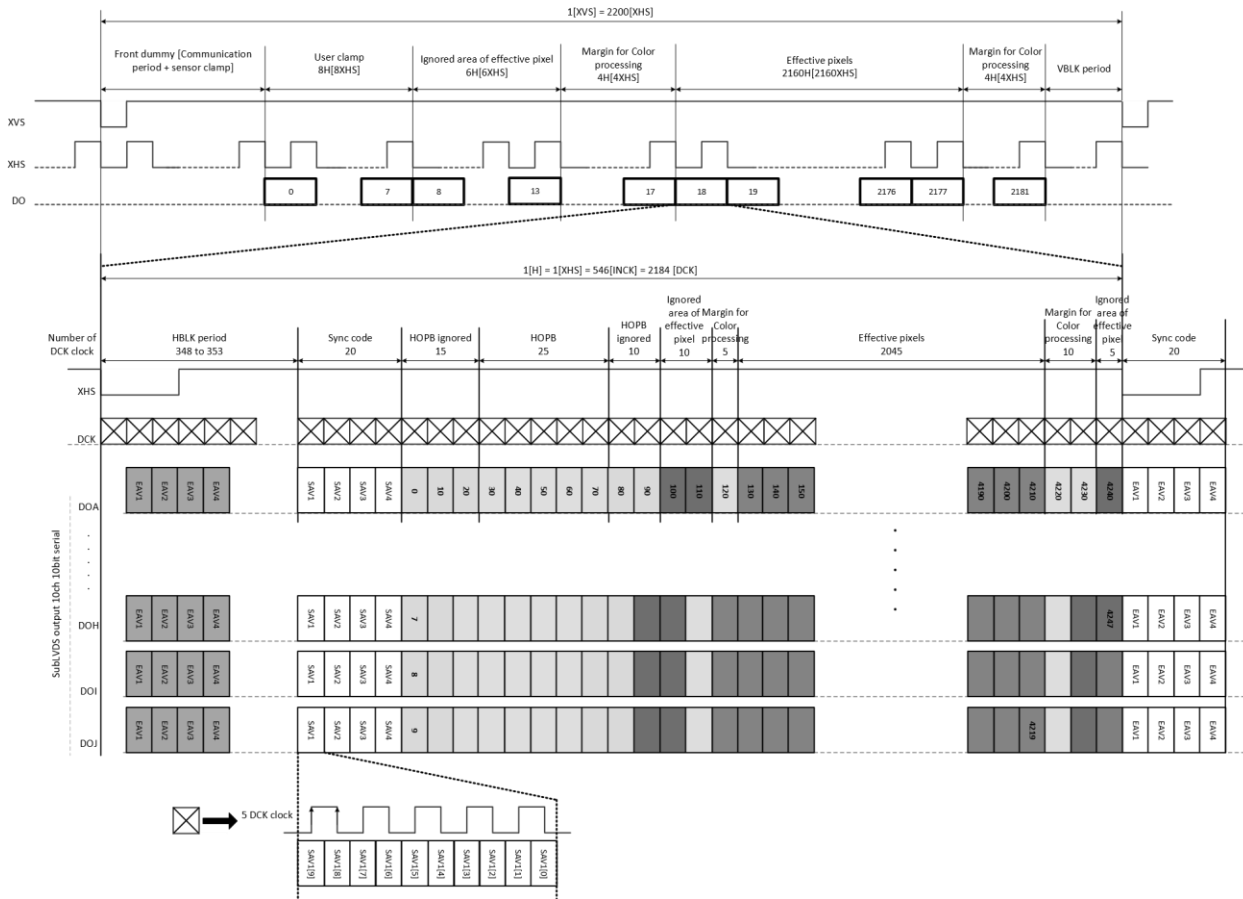


**Figure 2.8. SubLVDS Image Sensor Receiver IP Core Input Bus Waveform**

The horizontal and vertical timing of the received data are controlled by the XVS and XHS sync signals. The sync code is added before and after the pixel data. Table 2.6 lists the sync code details.

**Table 2.6. Sync Code Details**

| LVDS Output Bit No. | | Sync code | | | |
|---|---|---|---|---|---|
| **12-bit Output** | **10-bit Output** | **1st Word** | **2nd Word** | **3rd Word** | **4th Word** |
| 11 | 9 | 1 | 0 | 0 | 1 |
| 10 | 8 | 1 | 0 | 0 | 0 |
| 91 | 7 | 1 | 0 | 0 | V |
| 82 | 6 | 1 | 0 | 0 | H |
| 73 | 5 | 1 | 0 | 0 | P3 |
| 63 | 4 | 1 | 0 | 0 | P2 |
| 53 | 3 | 1 | 0 | 0 | P1 |
| 43 | 2 | 1 | 0 | 0 | P0 |
| 3 | 1 | 1 | 0 | 0 | 0 |
| 2 | 0 | 1 | 0 | 0 | 0 |
| 1 | — | 1 | 0 | 0 | 0 |
| 0 | — | 1 | 0 | 0 | 0 |

| | | Protection Bits | | | |
|---|---|---|---|---|---|
| **V** | **H** | **P3** | **P2** | **P1** | **P0** |
| 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 0 | 1 |
| 1 | 0 | 1 | 0 | 1 | 1 |
| 1 | 1 | 0 | 1 | 1 | 0 |

**Notes:**

1.  1: Blanking line; 0: Except blanking line
2.  1: End sync code; 2: Start sync code
3.  Protection bits

## 2.6. Sample Configurations

Waveforms below show the output behavior with different *Word Count* and *Dropped Pixel Count* but having same *Number of Rx Lanes* == 4, Data Type == RAW10 and number of pixels sent by sensor is 40 pixels.
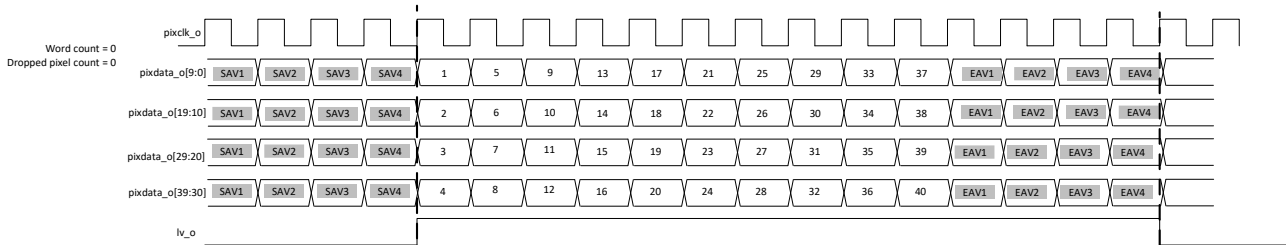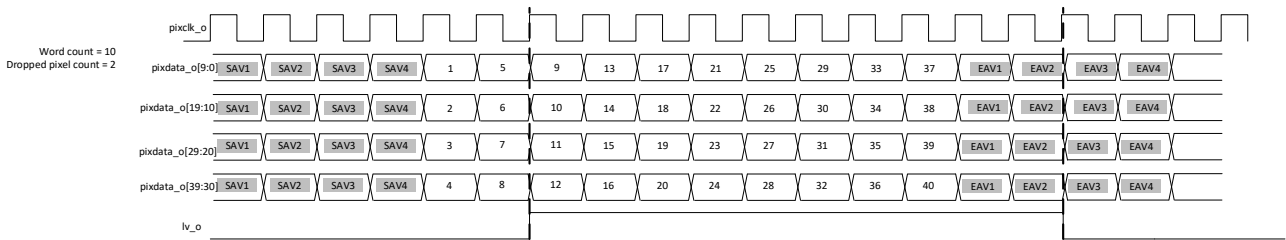


**Figure 2.9. SubLVDS Image Sensor Receiver Output Concept Waveform when Word Count == 0 and Dropped Pixel Count ==0**
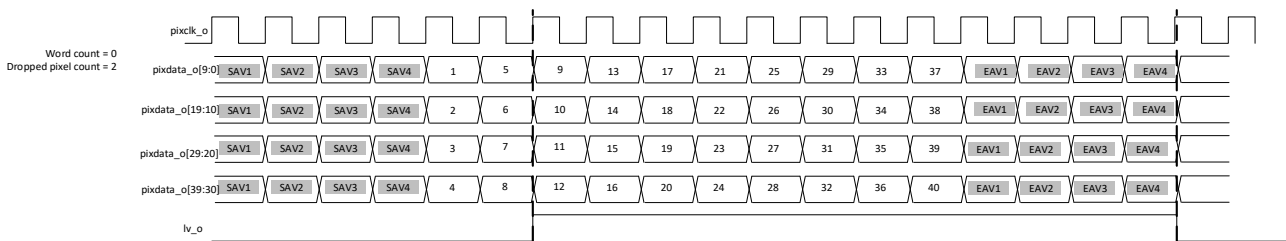


**Figure 2.10. SubLVDS Image Sensor Receiver  Output Concept Waveform when Word Count == 0 and Dropped Pixel Count ==2**
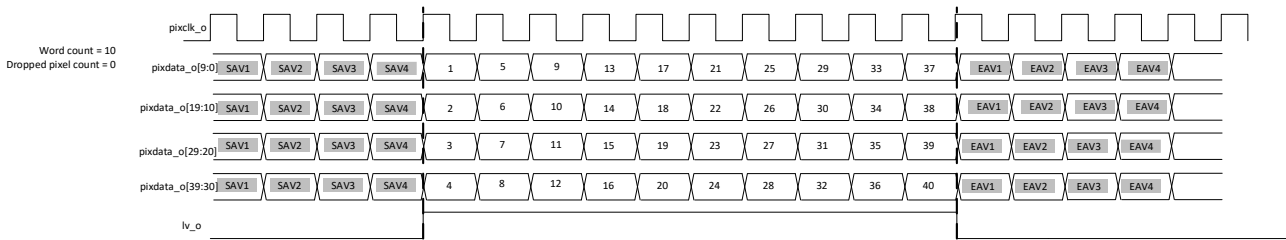


**Figure 2.11. SubLVDS Image Sensor Receiver  Output Concept Waveform when Word Count == 10 and Dropped Pixel Count ==0**



**Figure 2.12. SubLVDS Image Sensor Receiver  Output Concept Waveform when Word Count == 10 and Dropped Pixel Count ==2**

# 3.    IP Generation and Evaluation

This section provides information on how to generate the SubLVDS Image Sensor Receiver IP Core using the Lattice Radiant software and how to run simulation and synthesis. For more details on the Radiant software, refer to the Lattice Radiant Software 2.1 User Guide.

## 3.1.  Licensing the IP

An IP core-specific license string is required to enable full use of the SubLVDS Image Sensor Receiver IP Core in a complete, top-level design. You can fully evaluate the IP core through functional simulation and implementation (synthesis, map, place and route) without an IP license string. This IP Core supports Lattice's IP hardware evaluation capability, which makes it possible to create versions of the IP core, which operate in hardware for a limited time (approximately four hours) without requiring an IP license string. See Hardware Evaluation section for further details. However, a license string is required to enable timing simulation and to generate bitstream file that does not include the hardware evaluation timeout limitation.

## 3.2.  Generation and Synthesis

The Lattice Radiant software allows you to generate and customize modules and IPs and integrate them into the device's architecture. The procedure for generating SubLVDS Image Sensor Receiver IP Core in Lattice Radiant software is described below.

To generate the SubLVDS Image Sensor Receiver IP Core:

1.  Create a new Lattice Radiant software project or open an existing project.

2.  In the **IP Catalog** tab, double-click on **SubLVDS_Image_Sensor_Receiver** under **IP, Audio_Video_Image_Processing** category. The **Module/IP Block Wizard** opens as shown in Figure 3.1. Enter values in the **Instance name** and the **Create in** fields and click **Next**.
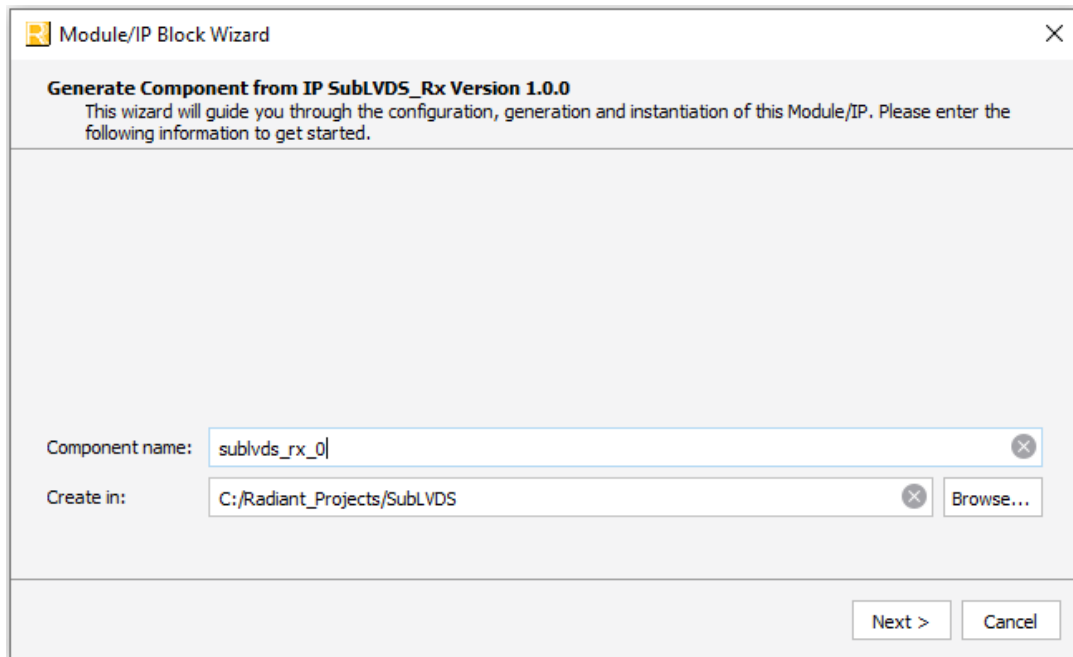


**Figure 3.1. Module/IP Block Wizard**

3.  In the modules dialog box of the **Module/IP Block Wizard** window, customize the SubLVDS Image Sensor Receiver IP Core according to custom specifications, using drop-down menus and check boxes. As a sample configuration, see Figure 3.2. For configuration options, see the Attribute Summary section.
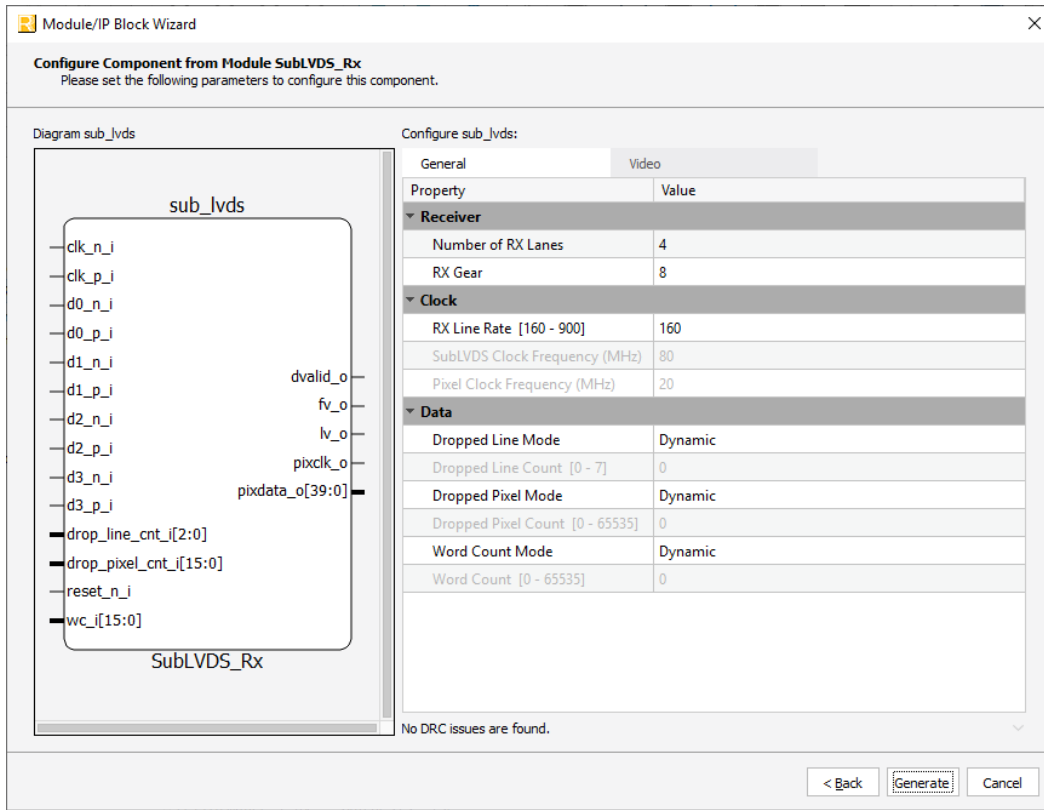
**Figure 3.2. Configure Block of SubLVDS Image Sensor Receiver IP Core**

4. Click **Generate**. The **Check Generated Result** dialog box opens, showing design block messages and results as shown in Figure 3.3.
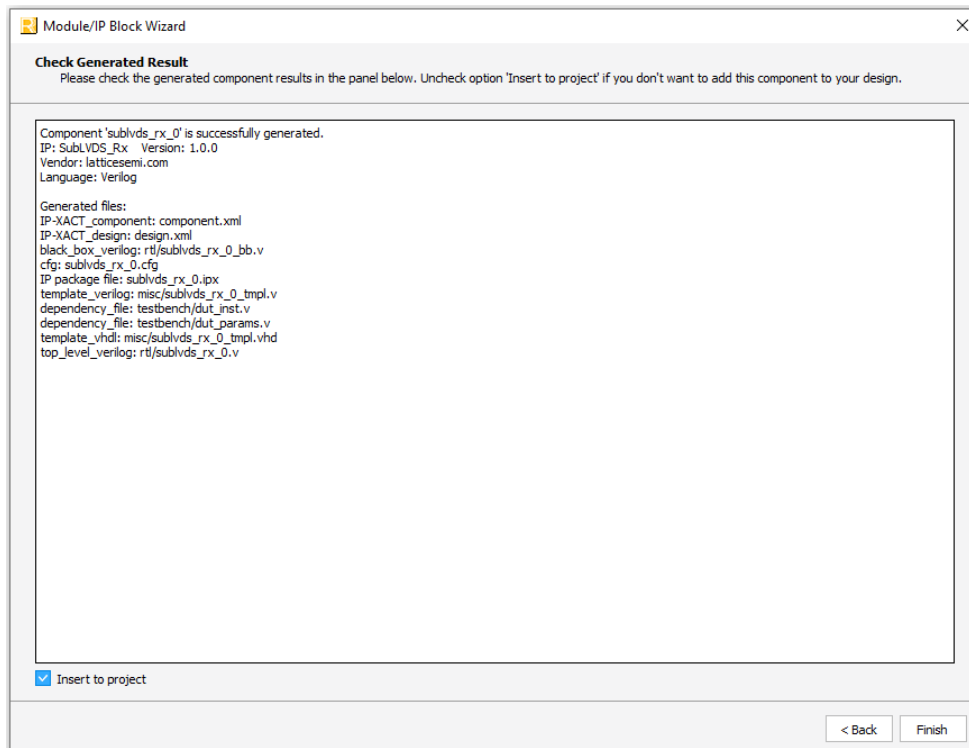


**Figure 3.3. Check Generated Result**

5. Click the **Finish** button. All the generated files are placed under the directory paths in the **Create in** and the **Instance name** fields shown in Figure 3.1.

The generated SubLVDS Image Sensor Receiver IP Core package includes the black box (<Instance Name>_bb.v) and instance templates (<Instance Name>_tmpl.v/vhd) that can be used to instantiate the core in a top-level design. An example RTL top-level reference source file (<Instance Name>.v) that can be used as an instantiation template for the IP core is also provided. You may also use this top-level reference as the starting template for the top-level for their complete design. The generated files are listed in Table 3.1.

**Table 3.1. Generated File List**

| Attribute | Description |
|---|---|
| <Instance Name>.ipx | This file contains the information on the files associated to the generated IP. |
| <Instance Name>.cfg | This file contains the parameter values used in IP configuration. |
| component.xml | Contains the ipxact:component information of the IP. |
| design.xml | Documents the configuration parameters of the IP in IP-XACT 2014 format. |
| rtl/<Instance Name>.v | This file provides an example RTL top file that instantiates the IP core. |
| rtl/<Instance Name>_bb.v | This file provides the synthesis black box. |
| misc/<Instance Name>_tmpl.v misc /<Instance Name>_tmpl.vhd | These files provide instance templates for the IP core. |

## 3.3. Running Functional Simulation

Running functional simulation can be performed after the IP is generated. The following steps can be performed.

1. Click the button located on the **Toolbar** to initiate the **Simulation Wizard** shown in Figure 3.4.
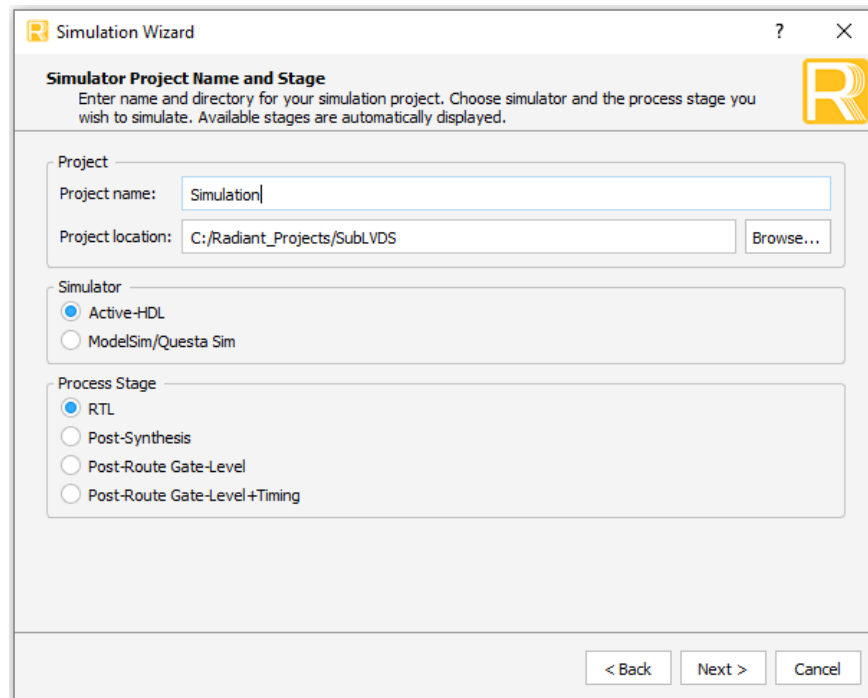


**Figure 3.4. Simulation Wizard**

2.  Click **Next** to open the **Add and Reorder Source** window as shown Figure 3.5.
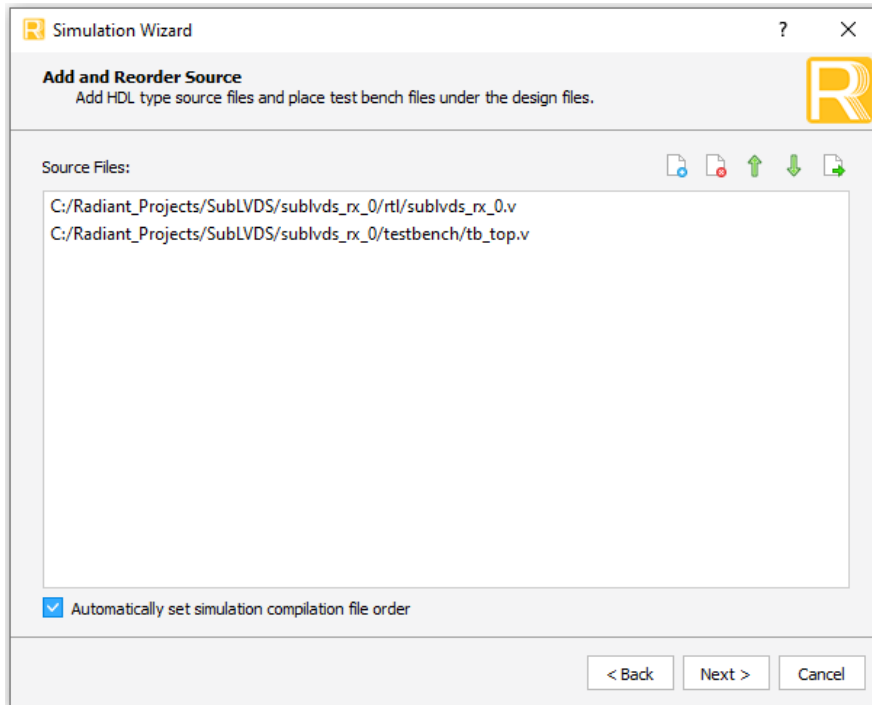


**Figure 3.5. Adding and Reordering Source**

3.  Click **Next**. The Summary window is shown. Click **Finish** to run the simulation.

    **Note:** It is necessary to follow the procedure above until it is fully automated in the Lattice Radiant software suite.

    The results of the simulation in our example are provided in Figure 3.6.
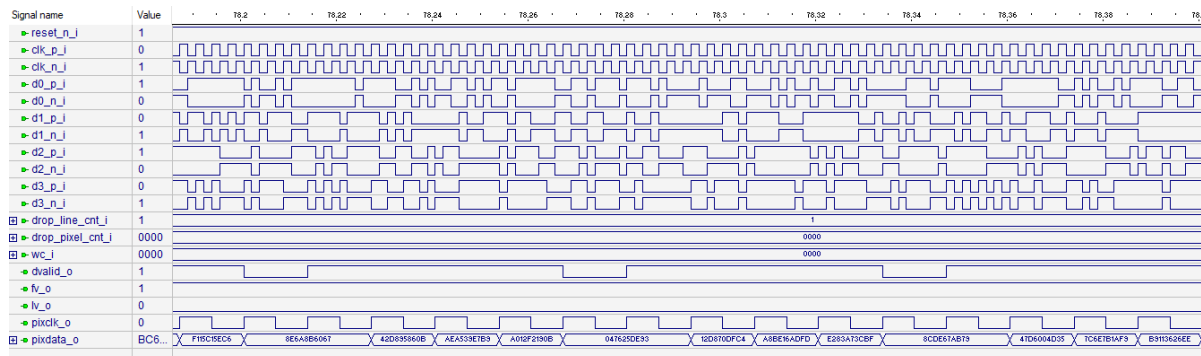


**Figure 3.6. Simulation Waveform**

## 3.4.  Hardware Evaluation

The SubLVDS Image Sensor Receiver IP Core supports Lattice's IP hardware evaluation capability when used with LIFCL devices. This makes it possible to create versions of the IP core that operate in hardware for a limited period of time (approximately four hours) without requiring the purchase of an IP license. It may also be used to evaluate the core in hardware in user-defined designs. The hardware evaluation capability may be enabled/disabled in the Strategy dialog box. It is enabled by default. To change this setting, go to Project > Active Strategy > LSE/Synplify Pro Settings.

# 4. Ordering Part Number

The Ordering Part Number (OPN) for this IP Core are the following:

- LVDS-RX-CNX-U – Sub-LVDS Image Sensor Receiver for CrossLink-NX – Single Design License
- LVDS-RX-CNX-UT – Sub-LVDS Image Sensor Receiver for CrossLink-NX – Site License
- LVDS-RX-CTNX-U – Sub-LVDS Image Sensor Receiver for Certus-NX – Single Design License
- LVDS-RX-CTNX-UT – Sub-LVDS Image Sensor Receiver for Certus-NX – Site License

# Appendix A. Resource Utilization

**Table A.1. Device and Tool Tested**

|  | Value |
|---|---|
| **Lattice Radiant Software Version** | ng2.1 (for Windows) |
| **Device Used** | LIFCL-40-9BG400I |
| **Performance Grade** | 9_High-Performance_1.0V |
| **Synthesis Tool** | Synplify Pro (R) Q-2020.03LR, Build 134R, May  8 2020 |
|  | Lattice Synthesis Engine (LSE) |

**Note:** Some bits are clipped to accommodate the current configuration with the selected device.

**Table A.2. SubLVDS-RX Resource Utilization**

| Number of RX Gears | RX Gear | Line Rate | Synthesis Tool | Register | LUTs | $F_{max}$* |
|---|---|---|---|---|---|---|
| 4 | 8 | 160 Mbps | Synplify Pro | 282 | 623 | pixclk_o = 20 MHz |
|  |  |  |  |  |  | clk_p_i = 20 MHz |
|  |  |  | LSE | 361 | 611 | pixclk_o = 20 MHz |
|  |  |  |  |  |  | clk_p_i = 20 MHz |
| 10 | 8 | 625 Mbps | Synplify Pro | 556 | 1116 | pixclk_o = 78 MHz |
|  |  |  |  |  |  | clk_p_i = 20 MHz |
|  |  |  | LSE | 829 | 1155 | pixclk_o = 78 MHz |
|  |  |  |  |  |  | clk_p_i = 20 MHz |
| 4 | 16 | 160 Mbps | Synplify Pro | 764 | 1698 | pixclk_o = 10 MHz |
|  |  |  |  |  |  | clk_p_i = 20 MHz |
|  |  |  | LSE | 1137 | 1673 | pixclk_o = 10 MHz |
|  |  |  |  |  |  | clk_p_i = 20 MHz |
| 4 | 16 | 1250 Mbps | Synplify Pro | 764 | 1698 | pixclk_o = 78 MHz |
|  |  |  |  |  |  | clk_p_i = 20 MHz |
|  |  |  | LSE | 1137 | 1673 | pixclk_o = 78 MHz |
|  |  |  |  |  |  | clk_p_i = 20 MHz |

*\*Note:* The Fmax provided here is shown to give affirmation to the user that that target frequency for a certain bitrate is attainable. While it is possible that the maximum frequency could be higher than the one described below, the IP is bounded to limit it in order to maintain the user's desired configuration.

# References

For complete information on Lattice Radiant Project-Based Environment, Design Flow, Implementation Flow and Tasks, as well as on the Simulation Flow, see the Lattice Radiant Software 2.1 User Guide.

# Technical Support Assistance

Submit a technical support case through www.latticesemi.com/techsupport.

# Revision History

**Revision 1.3, June 2020**

| Section | Change Summary |
|---|---|
| All | Updated Lattice Radiant software user guide references to version 2.1 across the document. |
| Introduction | Added support for Certus-NX in Table 1.1. |
| Ordering Part Number | Updated this section. |
| Appendix A. Resource Utilization | Added this section. |

**Revision 1.2, April 2020**

| Section | Change Summary |
|---|---|
| Functional Description | Corrected maximum line rate in Table 2.2. |

**Revision 1.1, February 2020**

| Section | Change Summary |
|---|---|
| Introduction | • Updated Table 1.1 to add LIFCL-17 as targeted device.<br>• Removed section 1.3.2. Data Ordering and Data Types |
| Functional Description | • Added pixclk_o port and note to Table 2.1.<br>• Revised V_TOTAL values in Table 2.2.<br>• Changed column heading to Description and updated descriptions of Dropped Pixel Count and Word Count attributes in Table 2.3.<br>• Added Sample Configurations section. |
| IP Generation and Evaluation | Corrected interface item to *Check Generated Result*. |
| Appendix A. | Added table reference in introductory paragraph. |

**Revision 1.0, December 2019**

| Section | Change Summary |
|---|---|
| All | Initial release |

# X-ON Electronics

Largest Supplier of Electrical and Electronic Components

*Click to view similar products for* Development Software *category:*

*Click to view products by* Lattice *manufacturer:*

Other Similar products are found below :

RAPPID-560XBSW  RAPPID-567XFSW  DG-ACC-NET-CD  SRP004001-01  SW006021-1NH  SW163052  SYSWINEV21  Core429-SA  SW500006-HPA  CWP-BASIC-FL  W128E13  CWP-PRO-FL  AD-CCES-NODE-1  NT-ZJCAT1-EV4  CWA-BASIC-FL  RAPPID-567XKSW  CWA-STANDARD-R  SW89CN0-ZCC  CWA-LS-DVLPR-NL  CWA-STANDARD-FL  VDSP-21XX-PCFLOAT  RAPPID-563XMSW  IPS-EMBEDDED  SWR-DRD-L-01  SDAWIR-4532-01  MPROG-PRO535E  AFLCF-08-LX-CE060-R21  WS02-CFSC1-EV3-UP  SYSMAC-STUDIO-EIPCPLR  LIB-PL-PC-N-1YR-DISKID  LS1043A-SWSP-PRM  SW006026-COV  T3DSO2000A-MSO  TMCC160-LC-CoE  TMCC160-LC-CANOPEN  1120270005  1120270006  MIKROBASIC PRO FOR FT90X (USB DONGLE)  MIKROC PRO FOR AVR (USB DONGLE LICENSE)  MIKROC PRO FOR FT90X (USB DONGLE)  MIKROBASIC PRO FOR AVR (USB DONGLE LICEN  MIKROBASIC PRO FOR FT90X  MIKROC PRO FOR DSPIC30/33 (USB DONGLE LI  MIKROC PRO FOR FT90X  MIKROC PRO FOR PIC32 (USB DONGLE LICENSE  52202-588  MIKROPASCAL PRO FOR ARM (USB DONGLE LICE  MIKROPASCAL PRO FOR FT90X  MIKROPASCAL PRO FOR FT90X (USB DONGLE)  MIKROPASCAL PRO FOR PIC32 (USB DONGLE LI