NANEYE_DESERIALIZER

v02

IP core Documentation

Document History

Revision	Date	Description	Author
1.0	May 24, 2011	Initial version	Michael Heil
1.1	June 25, 2011	Figure 5 corrected	Michael Heil
1.2	April 24, 2015	Update due to new versions of the sensor and deserializer	Michael Heil
2.0	June 08, 2015	Chapter 3.8.3. corrected, other minor corrections, document version number set to 2.0	Michael Heil
2.1	December 15, 2015	Minor corrections	Michael Heil

Table of contents

List of figures

List of tables

1. Overview

The FPGA IP core NANEYE_DESERIALIZER is used as an interface between the AWAIBA NanEye image sensor and a custom FPGA design for processing the image data. It decodes and deserializes the NanEye's data stream and provides the pixel data over an easily to use parallel interface together with the corresponding line-valid (LVAL) and frame-valid (FVAL) signals. Adoption to changes in the sensor's frame rate is performed dynamically. The IP core is furthermore responsible for transferring the user applied sensor parameters (offset, gain, integration time) into the sensor's configuration register [1].

This IP core is intended to be used in Xilinx-FPGAs but with a few modifications (see 4.) it can be ported to other FPGAs as well. The image-sensor has to be connected to the IP core by using a LVDS receiver, which can be located inside or outside the FPGA. An external level translator is necessary to adapt the output voltage of the FPGA to the required input level of the sensor during its configuration phase (see Figure 6). The output-enable input of this level shifter is controlled by the NANEYE_DESERIALIZER IP core.

The following sections describe how this IP core should be embedded into a custom FPGA design. The second part of this document illustrates the internal operation of the NANEYE_DESERIALIZER.

Important: Though the NANEYE_DESERIALIZER supports not only NanEye2 sensors but also NanEye3, NanEye3 is not scope of this documentation. Please note that support for NanEye3 might be completely removed in future versions of the IP core.

1.1. IP core Symbol

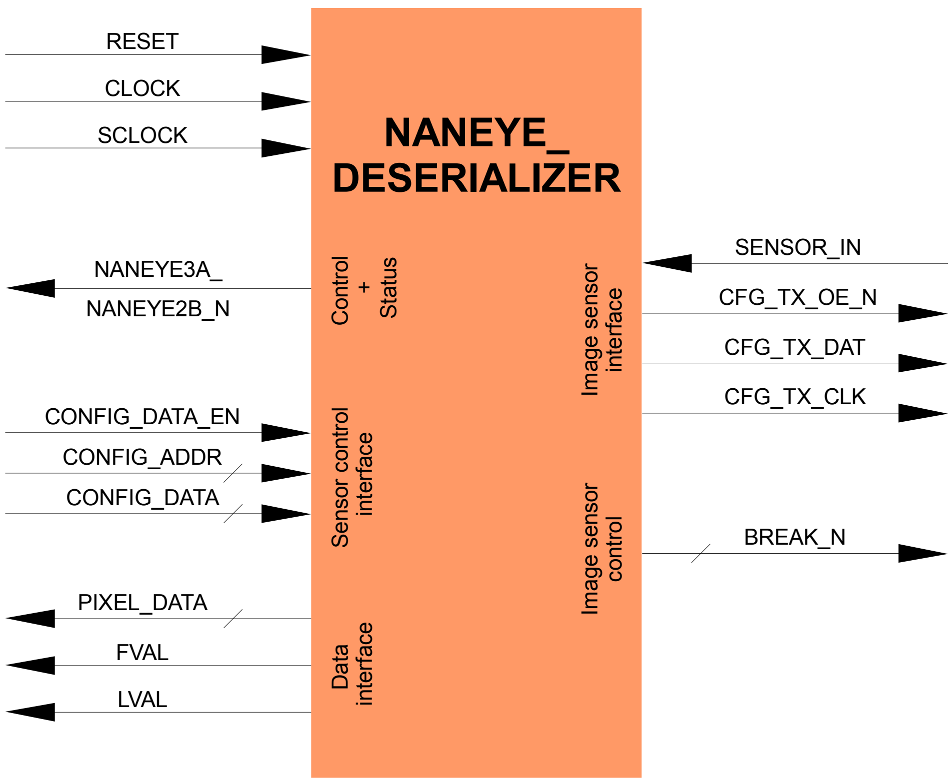

Figure 1: NANEYE_DESERIALIZER Symbol

1.2. I/O-Ports

Signal		I/O	Width	Description	Ref.
Group	**Name**				
System signals	RESET	I	1	Asynchronous reset, active-high	2.1.
	CLOCK 50MHz	I	1	System clock. The parameter G_CLOCK_PERIOD_PS has to be set to the actual CLOCK period in picoseconds	
	SCLOCK 200MHz	I	1	Sampling clock. The parameter G_SCLOCK_PERIOD_PS has to be set to the actual SCLOCK period in picoseconds	
Control Status signals	NANEYE3A_NANEYE2B_N	O	1	Indicates the type of sensor connected to this module: '0'=NANEYE2 '1'=NANEYE3	2.2.
Control interface	CONFIG_DATA_EN	I	1	'1' = data applied to CONFIG_DATA valid	2.3.
	CONFIG_ADDR	I	G_CADDR_W	Configuration register-address (has to be set to "000" for NanEye2)	
	CONFIG_DATA	I	G_CDATA_W	Configuration data for the sensor	
Data interface	PIXEL_DATA	O	G_PDATA_W	Pixel data output	2.4
	FVAL	O	1	Frame valid, active-high	
	LVAL	O	1	Line valid, active-high	
Image sensor interface	SENSOR_IN	I	1	Serial data from the sensor	2.5.
	CFG_TX_OE_N	O	1	Output enable for the external level translator, active-low	
	CFG_TX_DAT	O	1	Serialized configuration data	
	CFG_TX_CLK	O	1	Shift clock for the transferring the configuration data	
Image sensor control	BREAK_N	O	2	Controls voltage modulation (synchronous to SCLOCK)	2.6.
Debug	DEBUG_O	O	32	Not used	

Table 1: NANEYE_DESERIALIZER input and output signals

1.3. Parameters

Name	Type	Default value	Description
SIMULATION	boolean	false	Simulation mode yes/no
G_CLOCK_PERIOD_PS	integer	20833	CLOCK period in picoseconds
G_SCLOCK_PERIOD_PS	integer	5555	SCLOCK period in picoseconds
G_PDATA_W	integer	10	PIXEL_DATA width, has to be ≥ 10 and ≤ 16
G_CADDR_W	integer	3	CONFIG_ADDR width, must not be modified
G_CDATA_W	integer	16	CONFIG_DATA width, must not be modified

2. IP core usage

2.1. System signals

RESET is the asynchronous, active-high reset input of the module. The following table shows the reset states of the deserializer's output ports:

Output signal	**Output state when RESET = '1'**
NANEYE3A_NANEYE2B_N	'0'
PIXEL_DATA	"0000000000000000"
FVAL	'0'
LVAL	'0'
CFG_TX_OE_N	'1'
CFG_TX_DAT	'0'
CFG_TX_CLK	'0'
BREAK_N	"11"

Table 2: Reset states of output ports

CLOCK is the user interface clock input. The image data output interface signals (group "data interface", see 2.4.) and the sensor configuration input interface signals (group "control interface", see 2.3.) are synchronous / need to be synchronized to CLOCK. The parameter G_CLOCK_PERIOD_PS has to be set to the value of the CLOCK period in picoseconds. A clock frequency of 48 MHz or 50 MHz is recommended.

SCLOCK is the sampling clock which is used to sample the serial image data stream coming from the sensor (see 3.1.). Recommended clock frequency is >= 180 MHz. The parameter G_SCLOCK_PERIOD_PS has to be set to the value of the SCLOCK period in picoseconds.

The condition f(SCLOCK) > f(CLOCK) must be met.

2.2. Control / Status signals 从网上资料来看，Naneye 2B是2D的前辈，即原始版

NANEYE3A_NANEYE2B_N is a status output signal which indicates whether a NANEYE2 (NANEYE3A_NANEYE2B_N = '0') or a NANEYE3 (NANEYE3A_NANEYE2B_N = '1') sensor is connected.

2.3. Control interface

This is the user interface for the configuration of the NanEye sensor. These signals are used by the custom design to write data directly into the NanEye's configuration register to be able to modify integration time, gain, offset settings, etc. All inputs of this interface have to be synchronous to CLOCK.

The input vector CONFIG_ADDR is only of importance for using NanEye3. For usage with NanEye2, this input must be set to "000".

CONFIG_DATA (see Figure 2) corresponds directly to the content of the image-sensor's 16 bit configuration register [1].

CONFIG_DATA	Bits 15..14	Bits 13..12	Bits 11..10	Bits 9..8	Bits 7..0
Function	VREF_CDS[1..0] (NanEye2D)	VRST_PIXEL[1..0] (NanEye2D)	offset[2..1]	inverse_gain [2..1]	rows_in_reset [7..0]
	not used (NanEye2B/2C)	not used (NanEye2B/2C)			

Figure 2: CONFIG_DATA bits

After activation of CONFIG_DATA_EN for one clock cycle (see Figure 3) the value of the input vector CONFIG_DATA is sampled by the rising edge of CLOCK and stored into an internal 16-bit register.

This 16-bit value is internally expanded to a 24-bit vector by automatically adding the update-code ("1001"), the register-address ("000") and the reset-bit ("0") required by the sensor's protocol [1]. This 24-bit vector is then shifted out repeatedly to the sensor once per configuration-phase (phase number 252, according to [1]).

Important: As this interface doesn't provide a handshake mechanism, the user has to take into account, that an update of the configuration data is performed only once per frame. To be sure that the user applied CONFIG_DATA is actually written into the sensor's configuration register the time between two consecutive CONFIG_DATA_EN pulses should be longer than the reciprocal of the frame rate (see Figure 3).

Note: The reset value of the 24-bit shift register is 0x000000. This value is repeatedly sent to the sensor until CONFIG_DATA_EN is activated for the first time. As this reset value doesn't contain a valid update-code, the sensor ignores it.

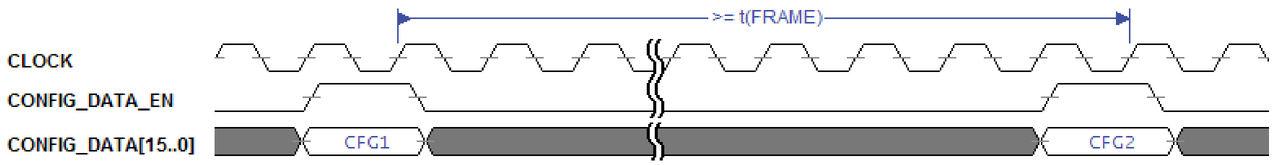

Figure 3: Timing diagram: Applying configuration data

2.4. Data interface

The data interface provides the NanEye's image data in parallel form together with the corresponding line-valid (LVAL) and frame-valid (FVAL) signals. The output signals of this interface are synchronous to CLOCK.

PIXEL_DATA is a G_PDATA_W bit wide output vector for the sensor's pixel data. Each pixel is represented as a 10-bit value. When G_PDATA_W > 10 the 10-bit pixel values are right aligned within this vector and the upper G_PDATA_W-10 bits are set to '0'.

FVAL is the frame valid signal. Each frame consists of 250 lines, i.e. while FVAL = '1', 250 LVAL pulses are given out.

LVAL is the line valid signal. One line consists of 250 columns, i.e. each LVAL pulse is 250 CLOCK cycles long.

Frames are given out, starting with the left uppermost pixel P(0,0) (see Figure 4).

Note: As the sensor actually transmits only 249 columns per line, the last (250th) column given out by the NANEYE_DESERIALIZER contains invalid pixels. As the last (250th) line of the sensor contains only 248 pixels, the last two columns (249th and 250th) of the 250th line given out by the NANEYE_DESERIALIZER contain invalid pixels.

The signal FVAL is actually activated several clock cycles before the rising edge of LVAL. This gives the custom design the possibility to prepare itself for the new frame (see Figure 5).

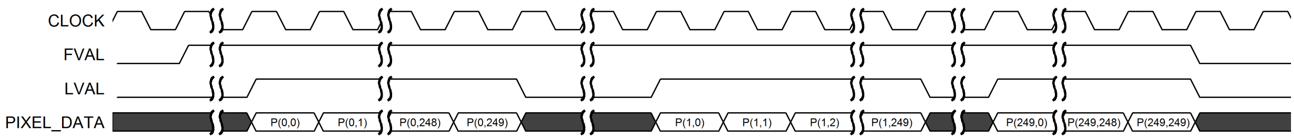

Figure 4: Timing: Reading out a complete frame

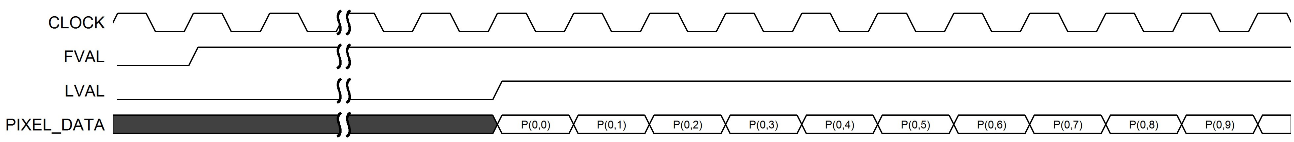

Figure 5: Timing: Frame start

2.5. Image sensor interface

The following simplified schematic (Figure 6) shows how to connect the NanEye sensor to the NANEYE_DESERIALIZER. Once per frame, after the last line of an image was completely transmitted, the sensor enters the configuration phase (phase number 252, according to [1]). During this phase the differential outputs of the sensor become high-impedance. This allows the NANEYE_DESERIALIZER to serially shift in the 16-bits of the configuration word (see 2.3., [1]) to change the sensor's settings for integration time, offset, etc. This 16-Bit word ("Register data") is transmitted within a simple 24-Bit protocol (see Figure 7). Level translators are necessary to translate the FPGA's I/O voltage (usually 3.3V) to the sensor's VCC (1.7V..2.4V). Each time the NANEYE_DESERIALIZER detects the start of the configuration phase, CFG_TX_OE_N is driven to '0' which activates the outputs of the voltage translators. Then the configuration data is shifted out at CFG_TX_DAT with the falling edge of CFG_TX_CLK (= shift clock). The frequency of CFG_TX_CLK is 2.5 MHz. CFG_TX_OE_N is driven low as long as possible to prevent the Nan Eye's data lines from floating (see Figure 7).

During the readout phase of the sensor, the LVDS receiver transforms the differential data to a single-ended signal, which is processed by the NANEYE_DESERIALIZER (see below).

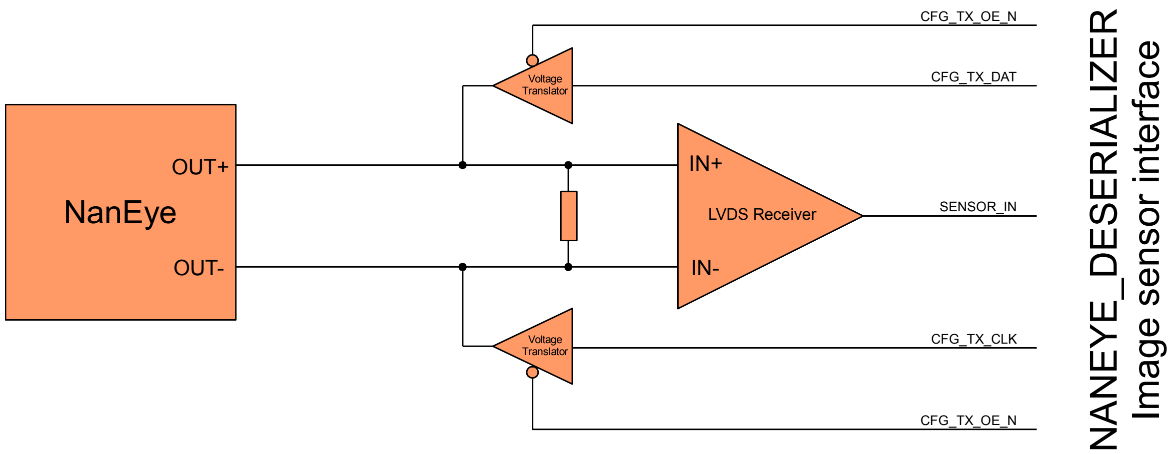

Figure 6: How to connect the NanEye2B to the NANEYE_DESERIALIZER

Note: The figure above represents only a simplified schematic. In reality the LVDS to single-ended conversion should be performed by a fast comparator and the common mode has to be adjusted with resistors as recommended in the NanEye's data sheet!

The following timing diagram shows the transmission of the value 0x058F (offset = 1, inverse_gain = 1, rows_in_reset = 0x8F) to the sensor's control register.

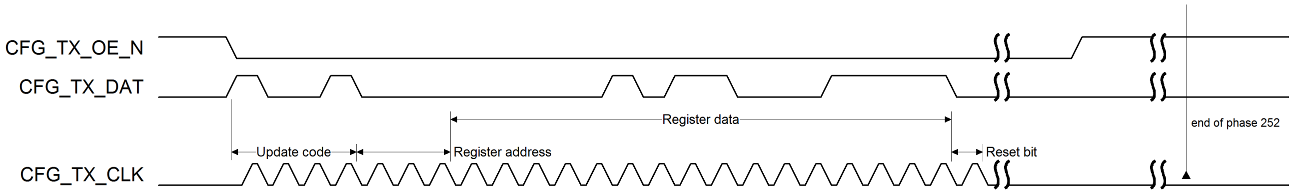

Figure 7: Timing diagram: Image sensor interface (configuration)

2.6. Image sensor control

The NanEye2D image sensor has an horizontal line artefact. The position of these lines depends on the sensor's integration time. It is possible to eliminate this artefact nearly completely by modulating the sensor's supply voltage appropriately instead of using a fixed power supply voltage. This modulation requires to apply a voltage V1 during the sensor's phase 252 and a voltage V2 during phase 253. During all other phases of the sensor, the nominal supply voltage V is applied. The NANEYE_DESERIALIZER provides the 2 bit output vector BREAK_N, which indicates when the phases 252 and 253 are active (see 3.9.). This vector can be used by an external circuit to control the supply voltage. The following table describes the meaning of BREAK_N:

Phase	BREAK_N[1..0]	Supply voltage
1.1..251	"11"	V
252	"10"	V1 = V - x
253	"01"	V2 = V - y
253a	"11"	V

Table 3: Voltage modulation

Please contact Awaiba regarding any information about the required values for V1 and V2!

3. Functional description

Figure 8 shows the internal structure of the IP core. The design mainly consists of two parts. One is responsible for the reception, decoding, deserialization and parallel output of the sensor's image-data, shown in the upper half of the block diagram. Data flows from left to right. The lower half of the block diagram depicts the data path for the transmission of the configuration data to the sensor's configuration register. Here the data flow is from right to left. The connection between both paths is necessary to synchronize the configuration circuit to the sensor's data stream and to find the configuration phase (phase number 252, i.e. high-impedance phase of the sensor's data lines). Furthermore the module LINE_PERIOD_CALC is responsible for determining the duration of phase 252 (see 3.8.).

For better clarity, the internal clock and reset connections are not shown within this block diagram. Simplified said, the sensor oriented side of the upper path is clocked by SCLOCK. The module RX_DECODER is additionally clocked by the inverted SCLOCK (double data rate sampling). The user-side (right side) as well as the module CONFIG_TX are clocked by CLOCK. The DPRAM separates both clock domains.

The NanEye image sensors transmit the image data as a Manchester-coded data stream, which is received and decoded by the module RX_DECODER. By continuously observing the sensor's data-stream the RX_DECODER is also able to derive the signals FRAME_START and CONFIG_EN, which are required for synchronization purposes. The succeeding block RX_DESERIALIZER contains a shift register which performs the serial-to-parallel conversion. It is also responsible for detecting the line-sync phases at the end of each received line. The parallel data-output of RX_DESERIALIZER is directly connected to the write-side of the block DPRAM (true dual-port RAM), which performs the transition of the image-data to the user's clock-domain CLOCK. The size of the RAM is dimensioned to store two lines of the pixel array. DPRAM_WR_CTRL generates the required signals for writing (address and write-enable) into the DPRAM. Each time the last pixel of one line was stored into the DPRAM, the signal LINE_FINISHED is pulsed. This triggers the readout procedure within the module DPRAM_RD_CTRL which is connected to the read-side of DPRAM. DPRAM_RD_CTRL generates the read-addresses as well as the synchronisation signals for the image data interface (FVAL, LVAL). To improve the ouput-timing, the output of the RAM is registered using the block OUT_REG.

CONFIG_CTRL determines the moment for triggering a transmission of configuration data after the user has activated CONFIG_DATA_EN. This is achieved by observing the decoder's output CONFIG_EN. A serial transmission is initiated after activating the output TX_START. CONFIG_TX contains the shift-register for the parallel-to-serial conversion as well as the shift-clock and output-enable generation.

Note: The module-names of the blocks correspond to the names of the VHDL components not to the instance names. The depicted blocks called OUT_REG and CFG_DATA_EN_SYNC are not realized as VHDL components but are coded as top-level processes. The corresponding VHDL description of a module can be found in the file named "component-name.vhd". Unused inputs or outputs of the depicted modules are not shown within this block-diagram.

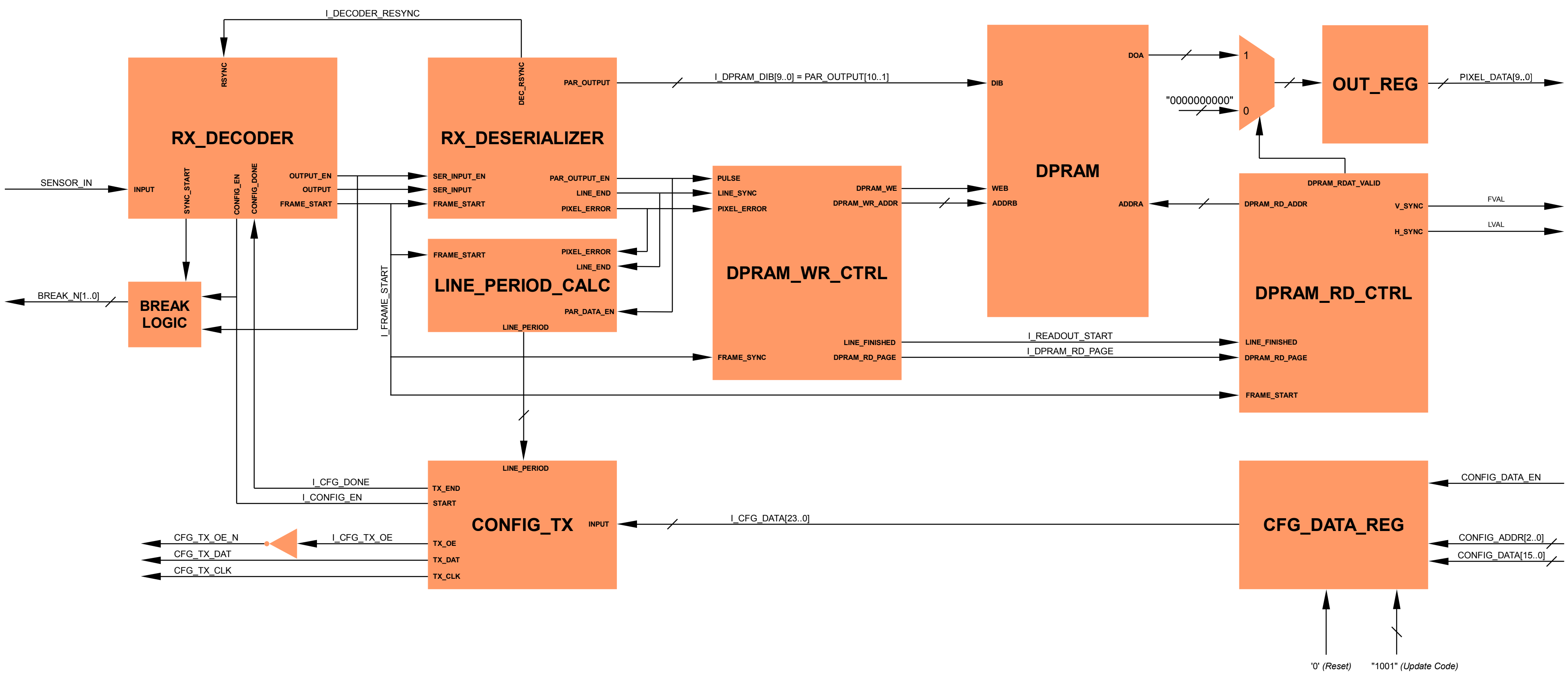

Figure 8: NANEYE_DESERIALIZER block diagram

3.1. RX_DECODER

3.1.1. I/O-Ports

Signal name	I/O	Width	Description
RESET	I	1	Asynchronous reset, active-high
CLOCK 200MHz	I	1	Clock-input, This input has to be connected to the sampling-clock SCLOCK
ENABLE	I	1	Module activation, active-high
RSYNC	I	1	Resynchronization of the decoder, active-high
INPUT	I	1	Input for the Manchester-coded data from the sensor
CONFIG_DONE	I	1	End of configuration phase, may be asynchronous to CLOCK, active-high
CONFIG_EN	O	1	Signals the start of the sensor's configuration phase, active-high
FRAME_START	O	1	Signals the start of frame, active-high
OUTPUT	O	1	Serial data output (decoded bits)
OUTPUT_EN	O	1	Signals valid data at OUTPUT, active-high
NANEYE3A_NANEYE2B_N	O	1	'0'= NanEye2 connected, '1'=NanEye3 connected
ERROR_OUT	O	1	Decoder error, active-high, not used
DEBUG_OUT	O	32	not used

Table 4: RX_DECODER input and output signals

3.1.2. Parameters

Name	Type	Default value	Description
G_CLOCK_PERIOD_PS	integer	5555	Decoder CLOCK period in picoseconds, see 2.1.

Table 5: RX_DECODER parameters

3.1.3. Functional description

This module performs the most important function within the NANEYE_DESERIALIZER as it is responsible for the reception end decoding of the Manchester-coded sensor data-stream [1]. Furthermore it detects and signals the regularly occurring events "start of frame" (FRAME_START = '1') and "start of configuration phase" (CONFIG_EN = '1') by continuously observing the input-signal.

3.1.3.1. Introduction

The Manchester-code represents the logical combination of serial data ("DATA") and its serial clock ("CLOCK") using the XOR-operation, see Figure 9. As a result, this signal changes its state at least once per clock-period. This allows the receiver to align correctly to the data-stream, without the need for the transmitter to send the clock signal along with the data signal. The data stream to be decoded by the receiver can be seen as a sequence of high or low pulses with only two possible pulse durations t_{short} = 0.5 * t_{Bit} (here also called "half-bit periods") and t_{long} = t_{Bit} (here also called "full-bit periods"). The NanEye's nominal shift clock frequency is f(CLOCK) ~ 30 MHz [1].

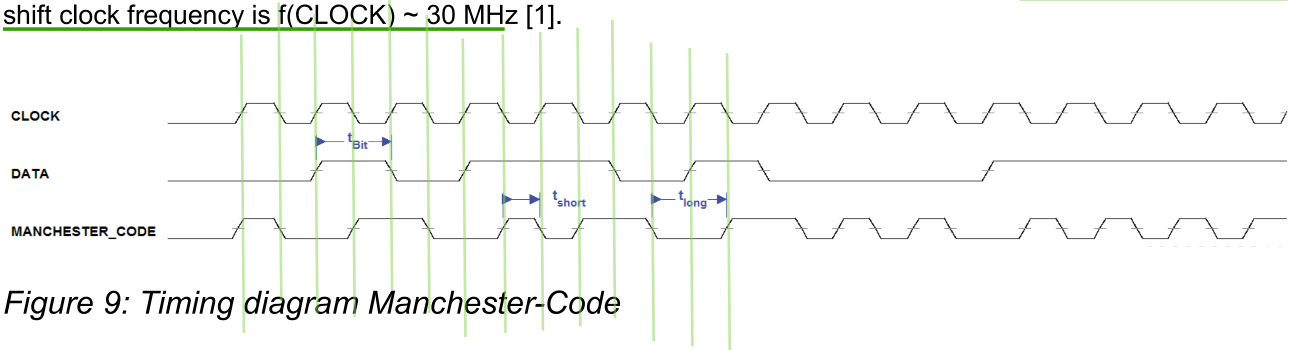

Figure 9: Timing diagram Manchester-Code

For decoding this signal, the receiver constantly needs to measure the time between two transitions. On the basis of this measurement the decoder has to classify whether the last transition was caused by a short pulse (S) or by a long pulse (L). A short pulse always means that the original data stream did not change its state, whereas a long pulse means that a transition occurred within the original bit-stream. Applying the Manchester-coded data stream of the example above (Figure 9) to the receiver should result in the sequence of pulses depicted in Figure 10. Each long pulse causes the decoded signal to change its state. If the initial state of the data-signal is known, the original bit-stream can be reconstructed as illustrated. In this example, as well as at the beginning of each line sent by the NanEye image sensor, the initial state of the original data signal is '0'.

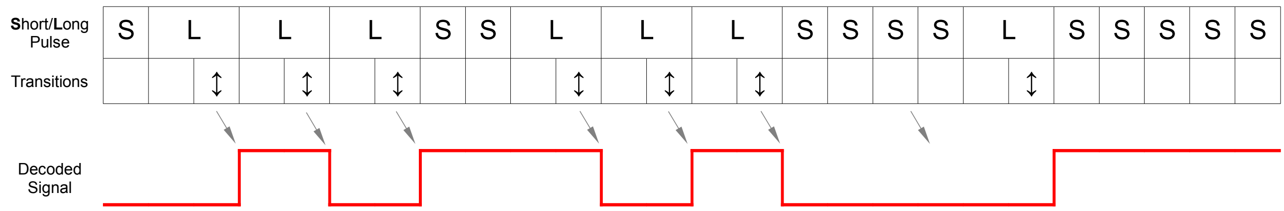

Figure 10: Decoding a Manchester-coded signal

3.1.3.2. Selecting the optimal sampling frequency

The image sensor's logic is driven by its own on-chip clock generator whose frequency is not exactly known, especially as it can drift by varying the operation conditions like temperature or supply voltage [1]. Therefore the NANEYE_DESERIALIZER has to sample the sensor's output signal as an asynchronous signal by using its own sampling clock (SCLOCK). This sampling is performed by RX_DECODER.

After this synchronization to the internal clock, the RX_DECODER is able to measure the duration of the pulses using a counter, which is clocked by the sampling clock as well. Due to the asynchronism between the sensor's clock and the sampling clock, the number of counted samples for a long pulse will not always be exact twice the number of samples counted for a short pulse. Additional effects like clock-jitter make the situation even worse. For this reason it is important to select a appropriate sampling frequency which guarantees that it is always possible to distinguish between long and short pulses by analyzing the counter values. On the other hand the selected frequency should not exceed the capabilities of a standard low-cost FPGA.

To be able to find the optimal sampling frequency, the following considerations were made: As can bee seen in Figure 9 the nominal time of a long pulse is t_{long}, the nominal time of a short pulse is t_{short} with $2*t_{short} = t_{long} = t_{Bit} = 1/f_{SERIAL\_CLOCK}$. The counter for measuring the pulse-duration is clocked by the sampling-clock whose frequency is $f_{SAMPLING\_CLOCK}$ which has to be higher than f_{SERIAL_CLOCK} (oversampling) to achieve an appropriate measurement resolution. The oversampling factor is $k = f_{SAMPLING\_CLOCK} / f_{SERIAL\_CLOCK}$. In the ideal case the counter measures a value of $CNT\_FB = t_{long} / k$ for a long pulse and a value of $CNT\_HB = t_{short} / k$ for a short pulse. Of course the relationship between t_{short} and t_{long} has to be valid for the counter values as well, i.e. $CNT\_FB = 2*CNT\_HB$.

Caused by the above mentioned effects, a pulse never will have the nominal duration. Instead the worst case has to be taken into account for estimating the minimum required sampling frequency. In this case a short pulse will be longer as normal (t_{short} + tolerance), so that the determined counter value will be higher than CNT_HB. An immediately following long pulse could be shorter as normal (t_{short} – tolerance) which causes the counter value to be lower than CNT_FB. To be able to distinguish between a short pulse and a long pulse CNT_HB always has to be smaller then CNT_FB.

The following graph (Figure 11) shows the resulting maximum tolerable tolerance in percent of the nominal pulse-duration in dependency on the oversampling factor k.

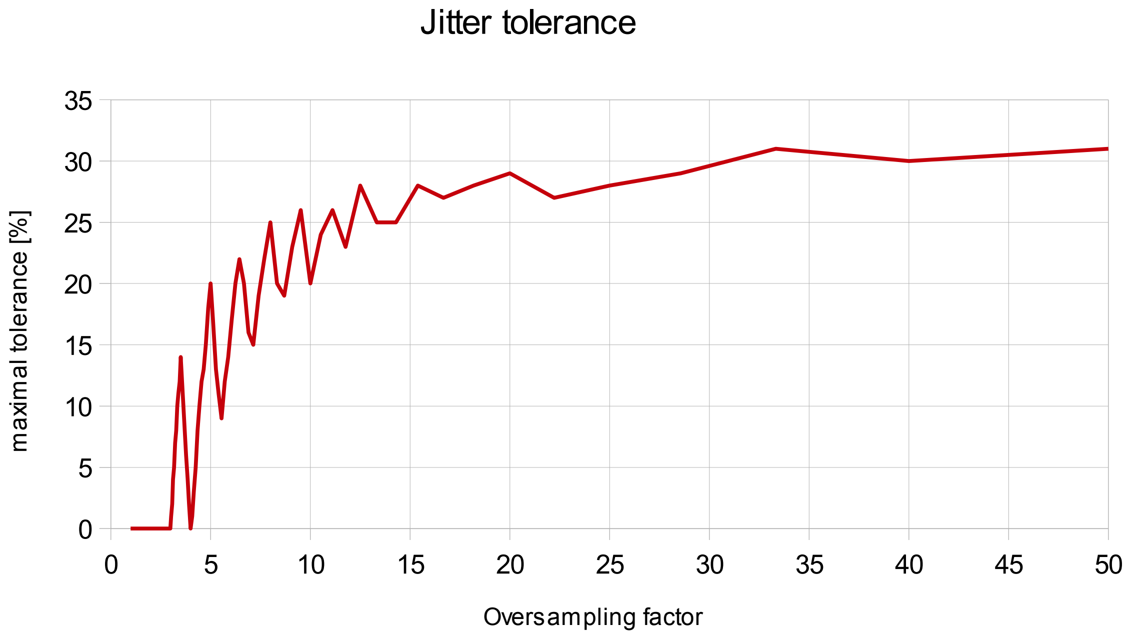

Figure 11: Sampling tolerance in dependency on the sampling frequency

Let's say the RX_DECODER uses the recommended sampling frequency of $f_{SAMPLING\_CLOCK}$ = 360 MHz (2*f(SCLOCK) = 2*180 MHz). Assuming a frequency for f_{SERIAL_CLOCK} = 36 MHz for the sensor's serial output ([1]) results in an oversampling factor of k = 360 MHz / 36 MHz = 10. So the nominal value for CNT_FB will be 10, for CNT_HB it will be 5.

The graph in Figure 11 shows for k = 10 a maximum allowable tolerance of 20%. This means that CNT_FB may be as short as CNT_FB = 8, and CNT_HB may be as long as CNT_HB = 6 and the RX_DECODER is still able to correctly identify them as long and short pulses as CNT_HB < CNT_FB.

Depending on the sensor's operating conditions (f_{SERIAL_CLOCK} is dependent on the sensor's VCC to allow to change the sensor's frame rate) f_{SERIAL_CLOCK} can vary between 30 MHz and 46 MHz according to [1]. Using a sampling clock frequency of 360 MHz results in a possible range for the oversampling factor of $7.8 \le k \le 12$. As the graph above shows, this sampling frequency is a good choice as it allows a reliable decoding with an possible tolerance of ~20 % over the complete frequency range of the sensor.

Note: Though it is theoretically possible to tolerate the deviations shown in the graph above, it is practically not possible to set a fixed threshold to distinguish between CNT_FB and CNT_HB. The main reason for this is the possibility to change the frame-rate (data-rate) of the NanEye sensor by varying the power supply voltage. Therefore the RX_DECODER uses the procedure described in 3.1.3.3. to dynamically adopt to the current data rate. This actually reduces the allowed tolerance but practical tests using this technique had shown good results.

3.1.3.3. Realization

The input stage of the RX_DECODER is formed by a double-data-rate register (IDDR, the Xilinx component is actually called IDDR2 [2]) which is clocked by CLOCK (= SCLOCK) and CLOCK_N (= inverted SCLOCK). Therefore the sensor's data-stream is read in by using an actual sampling frequency of 2*f(SCLOCK).

The 2-bit output-vector of this IDDR is fed through pipeline registers to meet the timing requirements. Using the outputs of these registers and with the help of two 5-bit adders it is possible to detect transitions in the input signal and to measure the time between these transitions. The determined time is stored within a 5-bit register whose output-vector is called I_BIT_LEN. This value represents the duration of the last pulse as a number of 2*f(SCLOCK) cycles. Each time a transition of the input signal occurs, i.e. the vector I_BIT_LEN was updated, the signal I_BIT_TRANS is activated (= '1').

At the beginning of each frame the RX_DECODER enters a calibration phase. This calibration phase is controlled by a FSM (**F**inite **S**tate **M**achine) called CAL_FSM (see Table 6) and is used to adopt the RX_DECODER dynamically to the current frame rate (which actually means to the current f_{SERIAL_CLOCK}). To do this, the FSM first of all searches the start of the sensor's phase 1.1 (=start of frame). Then it enters the state CAL_MEASURE in which a histogram of all measured I_BIT_LEN values over a complete frame is recorded. To save on logic resources and to meet timing requirements the later evaluation of the histogram is simplified by only considering entries which have reached a given threshold. This threshold is set to 2048. A 32 bit wide binary vector called I_HISTOGRAM_ENTRY_MAX holds the information of which histogram entry has reached the threshold. Figure 12 Illustrates this functionality.

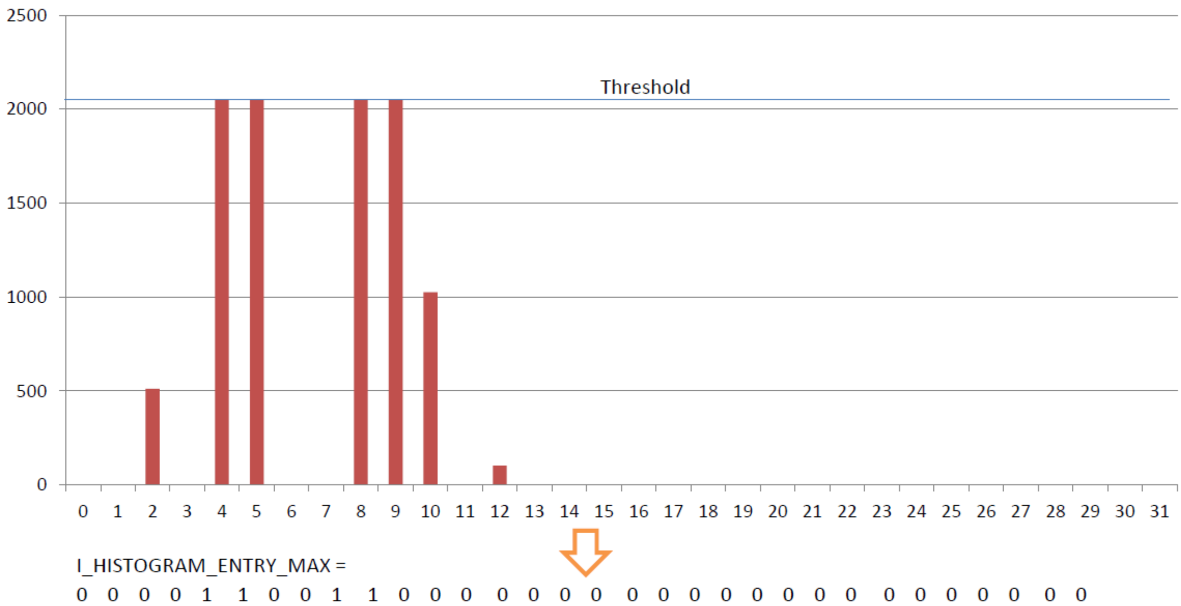

Figure 12: Data rate adaption with histogram in RX_DECODER

Next step is the determination of the minimum and maximum pulse durations (I_BIT_LEN_MIN and I_BIT_LEN MAX) of the just observed frame. Instead of evaluating the complete histogram, this task can now be simplified to checking only the vector I_HISTOGRAM_ENTRY_MAX. The values which are actually stored is the minimum value+1 for I_BIT_LEN_MIN (in the example above the value 5) and the maximum value–1 for I_BIT_LEN_MAX (in the example above the value 8). It is assumed that the changes of f_{SERIAL_CLOCK} between one frame and the subsequent frame is minimal, so the just calculated I_BIT_LEN_MIN / I_BIT_LEN_MAX values are always used for decoding the immediately subsequent frame.

As soon as the initial calibration (after reset deactivation) was performed (signalled by I_CAL_DONE = '1'), I_BIT_LEN_MIN and I_BIT_LEN_MAX represent valid pulse durations. Now, each time a signal transition occurs (I_BIT_TRANS is active), i.e. a new value for I_BIT_LEN is determined, I_BIT_LEN is compared with I_BIT_LEN_MIN and I_BIT_LEN_MAX. A pulse is considered to be a short pulse when its pulse duration is 'closer' to I_BIT_LEN_MIN and it's considered to be a long pulse when its pulse duration is 'closer' to I_BIT_LEN_MAX. Depending on this classification the signal I_HB_PERIOD ("half-bit period") or I_FB_PERIOD ("full-bit period") is activated for one clock cycle.

The signals I_HB_PERIOD and I_FB_PERIOD are the main inputs of a second FSM, called DECODE_FSM (see Table 7). This FSM constantly checks whether the sequence of short and long pulses is correct. In this case it gives the commands for inverting the state of the output signal each time a long pulse was detected. In case of a wrong sequence the module's output ERROR is pulsed ('1' for one clock-cycle). The RX_DECODER's input RSYNC forces this FSM into the state IDLE which causes the output signal to revert to its initial state '0'. The state of the output enable signal OUTPUT_EN is controlled by the current state of the DECODE_FSM as well. This signal is activated each time OUTPUT was updated, i.e. a bit was decoded and OUTPUT can be read by a connected module.

The calibration FSM (CAL_FSM) is also responsible for detecting the sensor's configuration phase (phase nbr. 252) and the start of a new frame. With the detection of the configuration phase, the RX_DECODER's output CONFIG_EN is activated (FSM state WAIT_FOR_SENSOR_CFG). Now the FSM waits for the reception of a CONFIG_DONE pulse. Next step is trying to find the start of a new frame. As soon as it was found (state CAL_SYNC_FOUND), the output signals SYNC_START and FRAME_START are activated for one clock cycle.

Figure 13 shows a simplified block diagram of most important parts of the RX_DECODER. The names printed in italics at the top side of the depicted blocks correspond to the names of the VHDL processes which describe the functionality of this block.

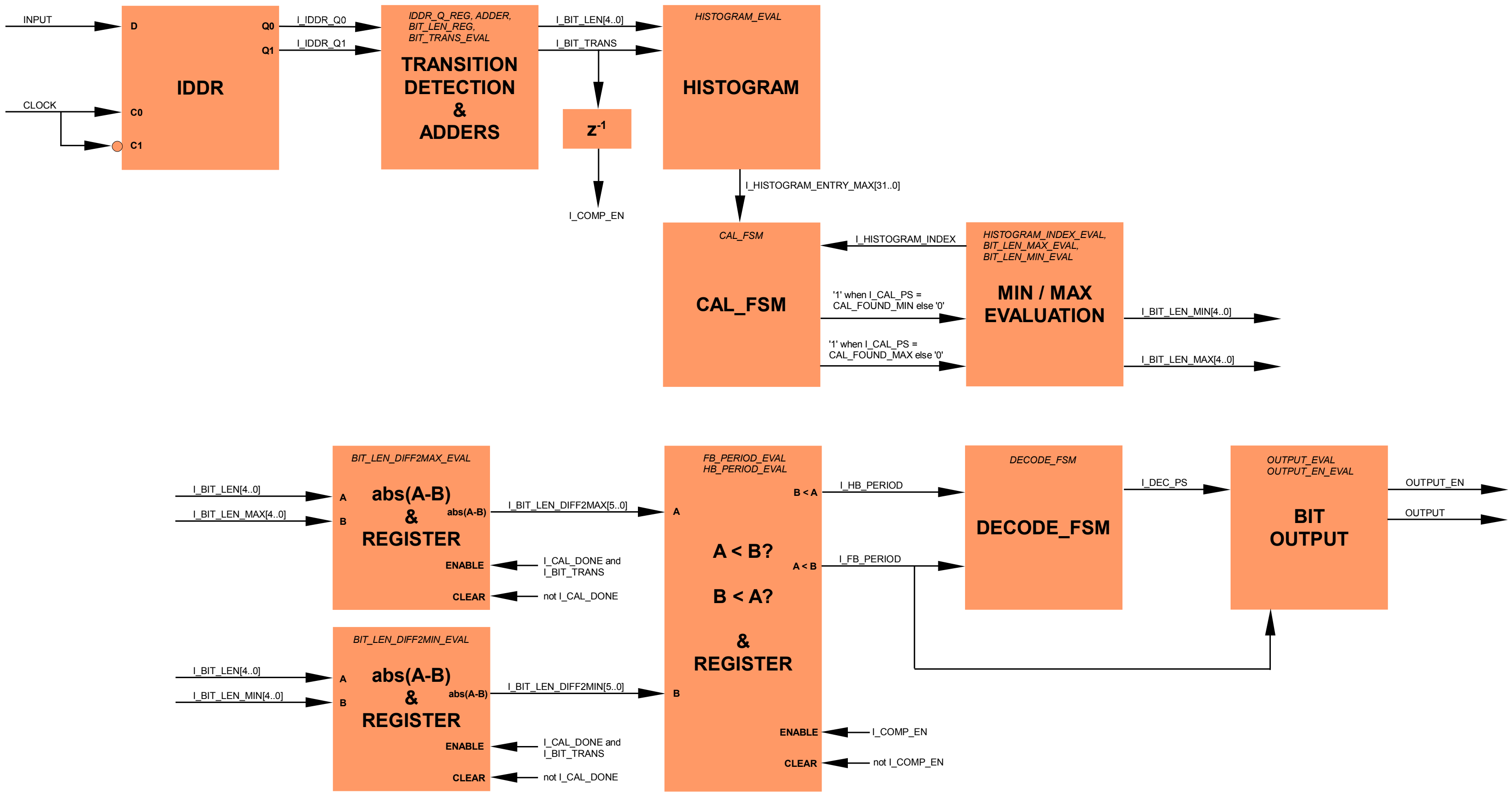

Figure 13: RX_DECODER: Simplified block diagram of the decoding function

Table 6 describes the functions and state transitions of the calibration-FSM (CAL_FSM). As can be seen, the states are more or less traversed linearly. Only the signals RESET and ENABLE are able to force the FSM from any state back to IDLE state. The states have the following meanings and conditions for switching to the next state:

State name	Description
CAL_IDLE	Reset state, after the first transition in the data stream is detected (I_BIT_TRANS = '1'), the FSM switches to CAL_FIND_FS.
CAL_FIND_FS	A longer phase without any transitions is considered to be the configuration phase (phase nbr. 252) and causes the FSM to switch to WAIT_FOR_SENSOR_CFG.
WAIT_FOR_SENSOR_CFG	Configuration data is transmitted to the sensor (by CONFIG_TX) and the FSM waits until it receives a high level at the module's input DONFIG_DONE, which signals the end of the serial transmission of configuration data. Then the FSM switches to CAL_FIND_SYNC.
CAL_FIND_SYNC	The FSM waits for the first transitions in the data stream. This is the sensor's phase 253 (see [1]) which is used for resynchronisation. Then the FSM changes to CAL_SYNC_FOUND.
CAL_SYNC_FOUND	The FSM remains for one clock cycle within this state before it switches to CAL_MEASURE. This state is used to generate the output signals SYNC_START and FRAME_START.
CAL_MEASURE	During this phase the histogram for the I_BIT_LEN values is generated. As soon the end of frame is detected the FSM starts to control the evaluation of the histogram by switching to the state CAL_SEARCH_MIN
CAL_SEARCH_MIN	This state is used for searching the shortest detected bit period (minimum value) within the vector I_HISTOGRAM_ENTRY_MAX. As soon as it was found the FSM transitions to CAL_FOUND_MIN
CAL_FOUND_MIN	Set I_BIT_LEN_MIN to minimum value + 1. Change to CAL_SEARCH_MAX
CAL_SEARCH_MAX	This state is used for searching the longest detected bit period (maximum value) within the vector I_HISTOGRAM_ENTRY_MAX. As soon as it was found the FSM transitions to CAL_FOUND_MAX
CAL_FOUND_MAX	Set I_BIT_LEN_MAX to maximum value + 1. Change to CAL_DONE
CAL_DONE	Measurement finished, the FSM switches immediately to the CAL_FIND_FS

Table 6: Description of the calibration-FSM

Table 7 describes the functions and state transitions of the decoder-FSM (DECODE_FSM). RESET = '1', ENABLE = '0' or RSYNC = '1' force the FSM from any state back to the state DEC_IDLE.

State name	Description
DEC_IDLE	FSM waits for the first frame start event. After it was found it switches immediately to DEC_START
DEC_START	FSM waits for the first full-bit period to synchronize, then it switches to DEC_SYNC
DEC_SYNC	FSM is successfully synchronized to the bit stream. Now it expects again a long-bit, which causes to stay within this state, or a short-bit which causes to switch to DEC_HALF_BIT
DEC_HALF_BIT	Within this state, the FSM expects only a short-bit, which causes the change to DEC_SYNC. Receiving a long-bit within this state causes an error (switch to DEC_ERROR)
DEC_ERROR	Generates an pulse on ERROR, the FSM switches immediately to the DEC_IDLE-state for resynchronization

Table 7: Description of the decoder-FSM

3.2. RX_DESERIALIZER

3.2.1. I/O-Ports

Signal name	I/O	Width	Description
RESET	I	1	Asynchronous reset, active-high
CLOCK	I	1	Clock-input
NANEYE3A_NANEYE2B_N	I	1	'0'= NanEye2 connected, '1'=NanEye3 connected, connected to corresponding output from RX_DECODER
FRAME_START	I	1	Input for the start-of-frame pulses, coming from RX_DECODER, active-high
SER_INPUT	I	1	Input for the decoded serial data (from RX_DECODER)
SER_INPUT_EN	I	1	Data on SER_INPUT valid (from RX_DECODER), active-high
DEC_RSYNC	O	1	Activates resynchronization of the decoder (to RX_DECODER), active-high
PAR_OUTPUT	O	12	Parallel data output
PAR_OUTPUT_EN	O	1	Data on PAR_OUTPUT valid, active-high
PIXEL_ERROR	O	1	Signals a start/stop bit error
LINE_END	O	1	Signals the end of each line, active-high
ERROR_OUT	O	1	Not used, set to '0'
DEBUG_OUT	O	16	not used

Table 8: RX_DESERIALIZER input and output signals

3.2.2. Parameters

Name	Type	Default value	Description
C_ROWS	integer	250	Number of rows, must be set to 250!
C_COLUMNS	integer	250	Number of columns, must be set to 250!

Table 9: RX_DESERIALIZER parameters

3.2.3. Functional description

The main component of this module is a 12-bit shift register which performs the serial-to-parallel conversion of the decoded bit sequence delivered by the RX_DECODER. The shift operation is enabled by SER_INPUT_EN = '1' and data is shifted in with the MSB first. A bit counter (I_BIT_CNT) is enabled (I_BIT_CNT_EN = '1') after the first pixel of a line with valid start- and stop-bits was received. With the help of this counter the moment of transferring the contents from the shift register to the parallel output register is determined (I_OUTREG_LOAD = '1') and the output pulses PAR_OUTPUT_EN are derived. PIXEL_ERROR is activated whenever a complete pixel was shifted in but the received values for the start-bit or/and the stop-bit do not match the expected values. This signal is internally (= I_PIXEL_ERROR) used to reset I_BIT_CNT_EN to '0', which as a consequence resets I_BIT_CNT to 0. Also the shift register and the column counter are reset to 0.

A further function of this module is to use the decoded bit stream to be able to detect the line- and frame boundaries by counting the number of columns (I_COL_CNT) and rows (I_ROW_CNT) delivered by the NanEye sensor. This knowledge is required to activate the signals DEC_RSYNC and LINE_END at the correct point of time. DEC_RSYNC is used within the decoder to reset the output signal to its initial value '0' at the beginning of each line (see 3.1.3.). LINE_END is required for controlling the read/write processes of the pixel values from/to the DPRAM (see Figure 8).

The above described function is realized with the help of a FSM (**F**inite **S**tate **M**achine) which is controlled by counters for determining the number of received bits (I_INPUT_EN_CNT), the number of pixels per line (I_COL_CNT) and the number of rows per frame (I_ROW_CNT).

Important to mention is that the sensor actually sends out only 249 pixels per line. The last line (250th) contains only 248 pixels. This is taken into account by the state machine. Furthermore the sensor transmits a short random bit sequence during phase 253a (see [1]). In case this sequence contains '1' bits (see Figure 14), this is detected by the RX_DESERIALIZER as an pixel error. This means, that I_PIXEL_ERROR is activated so that the above mentioned signals are reset and the deserialization is restarted. This means that the first line should be always correctly received.

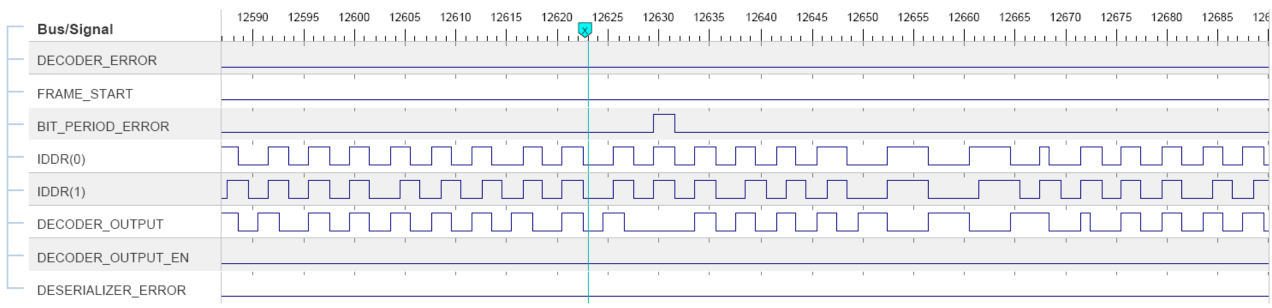

Figure 14: Faulty behaviour of the NanEye Sensor at the beginning of the first line

Table 10 describes the functions and state transitions of the FSM. RESET = '1' forces the FSM back to the IDLE-state.

State name	Description
IDLE	Reset state, FRAME_START = '1' causes the FSM to switch to FR_START
FR_START	Shift register, I_COL_CNT and I_ROW_CNT are reset FSM goes directly to LINE_VALID state
LINE_VALID	During this state, the FSM waits until one row was completely received, taking into account that the sensor only transmits 249 pixels per line and the last row contains only 248 pixels
RSYNC	An RSYNC pulse is output to resynchronize the RX_DECODER, the FSM switches immediately to LINE_VALID
LINE_VALID	During this state, the FSM waits until one row was completely received, taking into account that the sensor only transmits 249 pixels per line and the last row contains only 248 pixels. After one row was completely received, the FSM switches to LINE_SYNC, after the last pixel of a frame was received it switches to FRAME_END
LINE_SYNC	The FSM waits for the reception of the line-sync period and switches to INC_ROW_CNT, a LINE_END pulses is generated
INC_ROW_CNT	The row counter is incremented, the FSM switches immediately to LINE_VALID
FRAME_END	A LINE_END pulses is generated, the FSM switches immediately to IDLE

Table 10: Description of the RX_DESERIALIZER FSM

3.3. DPRAM_WR_CTRL

3.3.1. I/O-Ports

Signal name	I/O	Width	Description
RESET	I	1	Asynchronous reset, active-high
CLOCK	I	1	Clock-input
PULSE	I	1	Has to be connected to the output PAR_OUTPUT_EN of RX_DESERIALIZER
PIXEL_ERROR	I	1	Start/stop bit error from RX_DESERIALIZER
LINE_SYNC	I	1	Input for the end-of-line pulses, active-high
FRAME_SYNC	I	1	Input for the start-of-frame pulses, active-high
DPRAM_WR_ADDR	O	C_ADDR_W	Write address output for the DPRAM
DPRAM_WE	O	1	Write enable pulses for the DPRAM, active-high
DPRAM_RD_PAGE	O	1	Page select signal for DPRAM_RD_CTRL
LINE_FINISHED	O	1	Last pixel of a line was written to DPRAM, active-high

Table 11: DPRAM_WR_CTRL input and output signals

3.3.2. Parameters

Name	Type	Default value	Description
C_ADDR_W	integer	9	Address output width, must be set to 9

Table 12: DPRAM_WR_CTRL parameters

3.3.3. Functional description

Within the NANEYE_DESERIALIZER a DPRAM (true dual-port RAM) is used to separate the two clock-domains (SCLKOCK / CLOCK), which allows the user to read out the image data synchronously to the applied user input clock signal (CLOCK). The DPRAM is split into two pages, each one has the capacity for storing one line. While it is written into one page, the other one can be read out without disturbing the write-process. This process is completely transparent to the user.

DPRAM_WR_CTRL is responsible for controlling the write-side of the DPRAM and is therefore clocked by SCLOCK. The module generates the write-addresses as well as the write enable signal. The write address counter (I_DPRAM_WR_ADDR) is reset to 0 by the events end-of-line (LINE_SYNC = '1') or start-of-frame (FRAME_SYNC = '1') or a pixel error (PIXEL_ERROR = '1'). The write enable signal (DPRAM_WE) is derived by delaying the input signal PULSE. The write page signal (I_DPRAM_WR_PAGE) which toggles each time the last pixel of a line was written to the DPRAM forms the MSB of the write address. It is negated and given out as DPRAM_RD_PAGE, where it is used as MSB of the read address, to ensure that write and read accesses does never occur on the same page.

3.4. DPRAM

3.4.1. I/O-Ports

Signal name	I/O	Width	Description
CLKA	I	1	Clock input for port A
CLKB	I	1	Clock input for port B
ENA	I	1	Enable input for port A, active-high
ENB	I	1	Enable input for port B, active-high
WEA	I	1	Write enable for port A, active-high
WEB	I	1	Write enable for port B, active-high
ADDRA	I	A_WIDTH	Address for port A
ADDRB	I	A_WIDTH	Address for port B
DIA	I	D_WIDTH	Write data for port A
DIB	I	D_WIDTH	Write data for port B
DOA	O	D_WIDTH	Read data on port A
DOB	O	D_WIDTH	Read data on port B

Table 13: DPRAM input and output signals

3.4.2. Parameters

Name	Type	Default value	Description
A_WIDTH	integer	9	Address width, must be set to 9
D_WIDTH	integer	16	Data width, must be set to a value ≥ 10

Table 14: DPRAM parameters

3.4.3. Functional description

This module represents a true dual-port RAM whose vector sizes for data and addresses can be configured via VHDL generics (parameters A_WIDTH and D_WIDTH). Both sides (port A and port B) can be independently read and written using different clocks on each side. The module itself is coded using standard VHDL without instantiating primitives of a certain FPGA. The Xilinx synthesis tool XST recognises this description as a DPRAM and implements it by automatically instantiating a Xilinx BlockRAM.

Within the NANEYE_DESERIALIZER this DPRAM is used to separate the two clock-domains (SCLOCK / CLOCK), which allows the user to read out the image data synchronously to the applied user input clock signal (CLOCK). Port B is used as the write-side (synchronous to SCLOCK), whereas port A makes up the user-side for reading. The size of the DPRAM in bits is 2^{A_WIDTH} * D_WIDTH bits, which allows 2 complete lines to be stored for paging (see 3.3.3.). The unused MSBs (when D_WIDTH > 10) of the data-vector are set to '0'.

3.5. DPRAM_RD_CTRL

3.5.1. I/O-Ports

Signal name	I/O	Width	Description
RESET	I	1	Asynchronous reset, active-high
SCLOCK	I	1	Clock-input (sampling-clock)
CLOCK	I	1	Readout clock (user-clock)
NANEYE3A_NANEYE2B_N	I	1	'0'= NanEye2 connected, '1'=NanEye3 connected, connected to corresponding output from RX_DECODER
FRAMING_ERROR	I	1	connected to '0', not used
FRAME_START	I	1	Input for the start-of-frame pulses, active-high
LINE_FINISHED	I	1	Input for the end-of-line pulses, to be connected to the output LINE_FINISHED of DPRAM_WR_CTRL, active-high
DPRAM_RD_PAGE	I	1	Page select signal from DPRAM_WR_CTRL
DPRAM_RD_ADDR	O	C_ADDR_W	DPRAM read address
DPRAM_RDAT_VALID	O	1	Always '1'. Controls output multiplexor (see Figure 8)
H_SYNC	O	1	Horizontal synchronization = LVAL
V_SYNC	O	1	Vertical synchronization = FVAL

Table 15: DPRAM_RD_CTRL input and output signals

3.5.2. Parameters

Name	Type	Default value	Description
C_ROWS	integer	250	Number of rows per frame, must be set to 250
C_COLUMNS	integer	250	Number of columns per frame, must be set to 250
C_ADDR_W	integer	9	Address output width, must be set to 9

Table 16: DPRAM_RD_CTRL parameters

3.5.3. Functional description

The module DPRAM_RD_CTRL generates the read-addresses for the DPRAM and gives out the LVAL (H_SYNC) and FVAL (V_SYNC) signals synchronously to the user-clock (CLOCK). Together with the output register connected to the read-side of the DPRAM (see Figure 8 and 3.6.) LVAL (= H_SYNC) and FVAL (= V_SYNC) are correctly aligned to the pixel data output (PIXEL_DATA) and form the image data interface of the NANEYE_DESERIALIZER.

The main component of this module is a FSM (**F**inite **S**tate **M**achine) which is synchronized to the sensor's data-stream by receiving the signals FRAME_START (from RX_DECODER) and LINE_FINISHED (from DPRAM_WR_CTRL), see Figure 8. These signals, which are synchronous to SCLOCK, are first of all synchronized to CLOCK (= user-clock). The output timing is determined by the FSM with the help of two counters (I_COL_CNT and I_ROW_CNT). H_SYNC is activated every time the state LINE_VALID is reached. V_SYNC is activated after the reception of a FRAME_START pulse.

Note: The signal V_SYNC is activated approximately one line before H_SYNC is activated. This helps the receiver to prepare itself for the next frame.

Table 17 describes the functions and state transitions of the FSM. RESET = '1' or FRAME_START = '1' forces the FSM back to the state FRAME_BREAK.

State name	Description
FRAME_BREAK	Reset state, H_SYNC and V_SYNC inactive, the FSM waits for the first LINE_FINISHED = '1' after a FRAME_START was received, then V_SYNC is activated and the FSM switches to LINE_VALID.
LINE_VALID	During this state H_SYNC is active and the FSM waits until the column-counter (I_COL_CNT) reaches its end-value. Then it switches to LINE_BREAK. If it is the last line (row-counter I_ROW_CNT reached also its end-value), it switches to FRAME_BREAK.
LINE_BREAK	During this state H_SYNC is inactive and the FSM waits until LINE_FINISHED = '1' before it switches again to LINE_VALID.

Table 17: Description of the DPRAM_RD_CTRL FSM

3.6. OUT_REG

This top-level process describes a 10-bit parallel register which is connected to the read-side of the DPRAM and is clocked by the user-clock CLOCK. It improves the timing and could be located into the I/O-cells if the output-interface is directly connected to the I/Os of the FPGA.

3.7. CONFIG_TX

3.7.1. I/O-Ports

Signal name	I/O	Width	Description
RESET	I	1	Asynchronous reset, active-high
CLOCK	I	1	Clock-input
START	I	1	Trigger for the start of the transmission, pulse, active-high
LINE_PERIOD	I	16	Line period in number of CLOCK cycles
INPUT	I	C_NO_CFG_BITS	Configuration data input-vector
TX_END	O	1	Signals the end of the transmission, pulse, active-high
TX_DAT	O	1	Serial configuration data to sensor
TX_CLK	O	1	Shift clock output to sensor
TX_OE	O	1	Output enable for the voltage translators, active-high

Table 18: CONFIG_TX input and output signals

3.7.2. Parameters

Name	Type	Default value	Description
CLOCK_PERIOD_PS	integer	20833	CLOCK period in picoseconds, must be set to G_CLOCK_PERIOD_PS
BIT_PERIOD_NS	integer	400	TX_CLK period in nanoseconds, must be ≥ 400
C_NO_CFG_BITS	Integer	24	Defined in SENSOR_PROPERTIES_PKG.vhd, must not be modified

Table 19: CONFIG_TX parameters

3.7.3. Functional description

This module performs the parallel-to-serial conversion of the configuration data as well as the shift clock generation. Furthermore it controls the output-enables of the external voltage translators. The user applied configuration data is shifted out with the MSB first using the falling edge of the generated shift clock. The frequency of the shift clock frequency in MHz is 1000/BIT_PERIOD_NS (timing-diagram see Figure 7). Transmission is started after a START pulse was received. At the end of a transmission (= falling edge of TX_OE) a pulse is given out at TX_END. The input LINE_PERIOD represents the duration of the transmission of one sensor line as a number of CLOCK cycles. This value is determined by the module LINE_PERIOD_CALC (see 3.8.) and is used to activate TX_OE as long as possible, even after the actual data transmission was already finished. The purpose of this is to drive the sensor's data lines almost during the complete remaining part of the phase 252 to prevent them from floating.

3.8. LINE_PERIOD_CALC

3.8.1. I/O-Ports

Signal name	I/O	Width	Description
RESET	I	1	Asynchronous reset, active-high
CLOCK	I	1	User clock
SCLOCK	I	1	Sampling clock
FRAME_START	I	1	Frame start pulse from RX_DECODER
PAR_DATA_EN	I	1	Connect to PAR_DATA_EN from RX_DESERIALIZER
PIXEL_ERROR	I	1	Connect to PIXEL_ERROR from RX_DESERIALIZER
LINE_END	I	1	Connect to LINE_END from RX_DESERIALIZER
LINE_PERIOD	O	16	Line period output in number of CLOCK cycles, has to be directly connected to the corresponding input of CONFIG_TX

Table 20: LINE_PERIOD_CALC input and output signals

3.8.2. Parameters

Name	Type	Default value	Description
G_CLOCK_PERIOD_PS	integer	20833	CLOCK period in picoseconds, must be set to G_CLOCK_PERIOD_PS from top-level
G_LINE_PERIOD_MIN_NS	integer	50000	shortest possible duration in nanoseconds for one line, default value should not be modified
G_LINE_PERIOD_MAX_NS	Integer	120000	longest possible duration in nanoseconds for one line, default value should not be modified

Table 21: LINE_PERIOD_CALC parameters

3.8.3. Functional description

This module is used to determine the duration of the transmission of one sensor line. It is given out via LINE_PERIOD as a number of CLOCK cycles. This measurement is performed periodically always after a start of a new frame. The input signal FRAME_START coming from RX_DECODER is used to detect this event. PAR_DATA_EN and PIXEL_ERROR coming from RX_DESERIALIZER signal the start of the first line. The duration of this line is determined with the help of a counter, clocked by CLOCK. This counter is stopped by an LINE_END pulse which is sent by RX_DESERIALIZER. The counter value is compared with G_LINE_PERIOD_MIN_NS and G_LINE_PERIOD_MAX_NS and limited when necessary. The resulting value is registered and given out over LINE_PERIOD. This value is used by CONFIG_TX (see 3.7.) to determine how long the sensor's data lines have to be driven to prevent them from floating.

3.9. BREAK_LOGIC

This logic is responsible for the generation of the BREAK_N[1..0] output signals which can be used for supply voltage modulation to correct the shutter line artifact. The RX_DECODER's output signals CONFIG_EN and SYNC_START are used for generating BREAK_N[0], which in fact indicates the sensor's phase 252, as shown below:

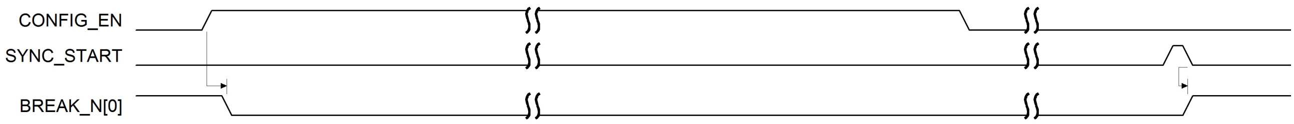

Figure 15: BREAK_N[0] Timing

DECODER_OUT_EN and SYNC_START of RX_DECODER are used to generate BREAK_N[1]:

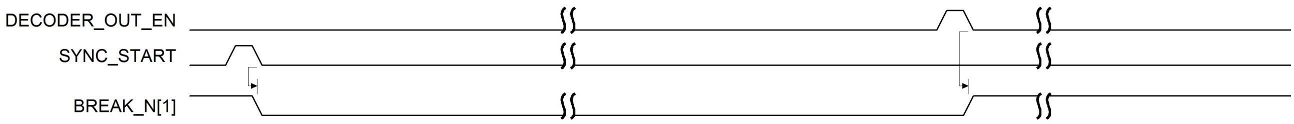

Figure 16: BREAK_N[1] Timing

4. Hardware requirements

4.1. Resources

The following table gives an example of the amount of required resources after implementing the NANEYE_DESERIALIZER as a top-level design into a Xilinx Spartan6 XC6SLX45-3fgg484

Device Utilization Summary				[-]
Slice Logic Utilization	**Used**	**Available**	**Utilization**	**Note(s)**
Number of Slice Registers	771	54,576	1%	
Number used as Flip Flops	771			
Number used as Latches	0			
Number used as Latch-thrus	0			
Number used as AND/OR logics	0			
Number of Slice LUTs	848	27,288	3%	
Number used as logic	838	27,288	3%	
Number using O6 output only	348			
Number using O5 output only	17			
Number using O5 and O6	473			
Number used as ROM	0			
Number used as Memory	0	6,408	0%	
Number used exclusively as route-thrus	10			
Number with same-slice register load	8			
Number with same-slice carry load	2			
Number with other load	0			
Number of occupied Slices	280	6,822	4%	
Number of MUXCYs used	500	13,644	3%	
Number of LUT Flip Flop pairs used	907			
Number with an unused Flip Flop	181	907	19%	
Number with an unused LUT	59	907	6%	
Number of fully used LUT-FF pairs	667	907	73%	
Number of unique control sets	54			
Number of slice register sites lost to control set restrictions	205	54,576	1%	
Number of bonded IOBs	74	316	23%	
IOB Flip Flops	1			
Number of RAMB16BWERs	0	116	0%	
Number of RAMB8BWERs	1	232	1%	

Number of BUFIO2/BUFIO2_2CLKs	0	32	0%	
Number of BUFIO2FB/BUFIO2FB_2CLKs	0	32	0%	
Number of BUFG/BUFGMUXs	2	16	12%	
Number used as BUFGs	2			
Number used as BUFGMUX	0			
Number of DCM/DCM_CLKGENs	0	8	0%	
Number of ILOGIC2/ISERDES2s	1	376	1%	
Number used as ILOGIC2s	1			

Table 22: Device utilization summary

List of literature

1: AWAIBA, NanEye 2D ASIC Specification, 2014
2: Xilinx, Spartan-3 Generation FPGA User Guide, v1.7