XILINX®

LogiCORE IP Image Noise Reduction v2.0

DS751 December 14, 2010 **Product Specification**

Introduction

The Xilinx Image Noise Reduction LogiCORE™ provides users with an easy-to-use IP block for reducing noise within each frame of video. The core has a programmable, edge-adaptive smoothing function to change the characteristics of the filtering in real-time.

Features

- Support for:
  - High-definition (1080p60) resolutions
  - Up to 4096 total pixels and 4096 total rows
- In-system update of smoothing filters
- Selectable processor interface
  - EDK pCore
  - General Purpose Processor
  - Constant Interface
- Support for 8-, 10-, or 12-bit input and output precision
- YCrCb or YUV 444 input and output
- For use with Xilinx CORE Generator™ software 12.4 or later
- Xilinx Streaming Video Interface (XSVI) bus simplifies connecting to other video IP

LogiCORE IP Facts Table						
Core Specifics						
Supported Device Family[1]	Spartan®-3A DSP, Spartan-6, Virtex®-5, Virtex-6					
Supported User Interfaces	General Processor Interface, EDK PLB 4.6, Constant Interface					
	Resources[2]					**Frequency**
Configuration	**LUTs**	**FFs**	**Slices**	**Block RAMs**	**DSP48s**	**Clock Freq. (MHz)[3]**
Data Width = 8	1226	1453	469	1(36K)+ 1(18K)	3	250
Data Width = 10	1409	1723	475	2(36K)	3	266
Data Width = 12	1622	1993	494	2(36K)	3	244
Provided with Core						
Documentation	Product Specification					
Design Files	Netlists, EDK pCore files, C drivers					
Example Design	Not Provided					
Test Bench	Not Provided					
Constraints File	Not Provided					
Simulation Model	Verilog and VHDL					
Tested Design Tools						
Design Entry Tools	CORE Generator™ tool, Platform Studio (XPS)					
Simulation	ModelSim v6.5c, Xilinx ISim 12.4					
Synthesis Tools	XST 12.4					
Support: Provided by Xilinx, Inc.						

1. For a complete listing of supported devices, see the release notes for this core.
2. Resources listed here are for Virtex-6® devices. For more complete device performance numbers, see Core Resource Utilization and Performance.
3. Performance numbers listed are for Virtex-6 FPGAs. For more complete performance data, see Core Resource Utilization and Performance.

Applications

- Pre-processing Block for Image Sensors
- Video Surveillance
- Video Conferencing
- Video Capture Devices

This core performs noise reduction by applying a smoothing filter to the image as show in Figure 1.

The smoothing filter is applied with a gain `K` which is dependent on the edge content in the image. Near edges, the gain is low so that the edges have less smoothing applied.

There is a choice of four different smoothing filters. The filters are of increasing strength in terms of the smoothing they provide. There is also an option to bypass the smoothing filter. The filter coefficients and frequency responses are shown in order of increasing strength in Figure 2, Figure 3, Figure 4, and Figure 5.

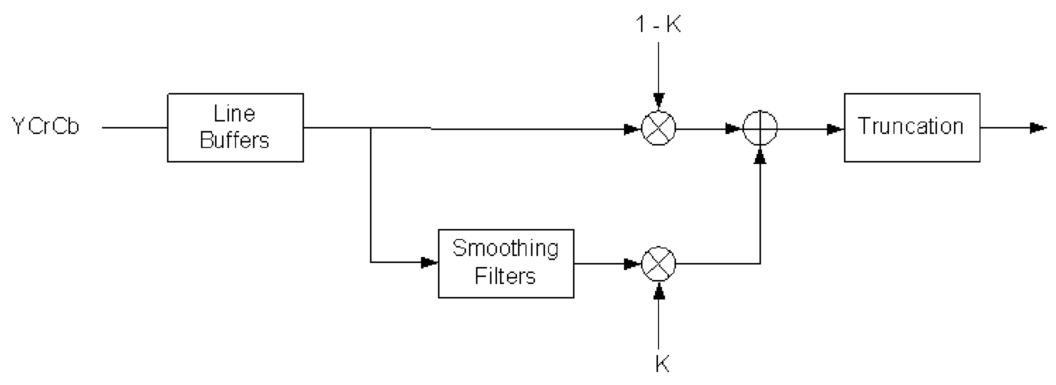


Figure 1: **Image Noise Reduction**

$$\text{Filter 1} = \begin{bmatrix} 121 \\ 161 \\ 121 \end{bmatrix}$$

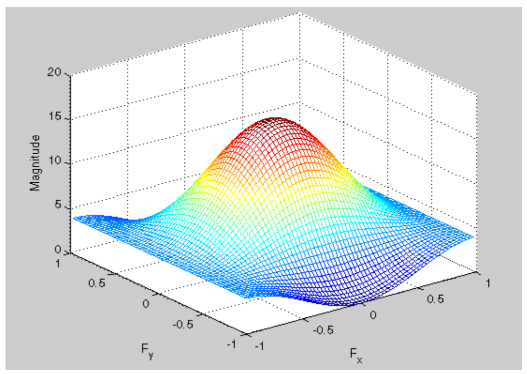


Figure 2: **Coefficients and Frequency Response for Filter Strength 1**

$$\text{Filter 2}=\begin{bmatrix}121\\242\\121\end{bmatrix}$$

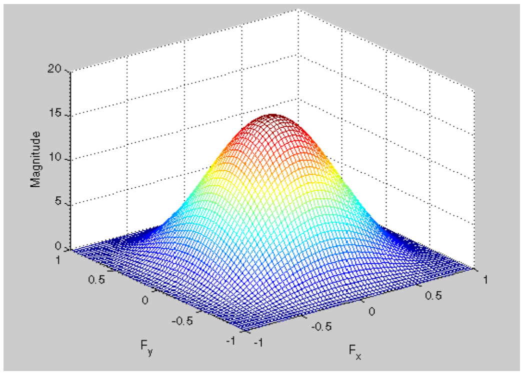


Figure 3: **Coefficients and Frequency Response for Filter Strength 2**

$$\text{Filter 3}=\begin{bmatrix}01110\\12421\\01110\end{bmatrix}$$

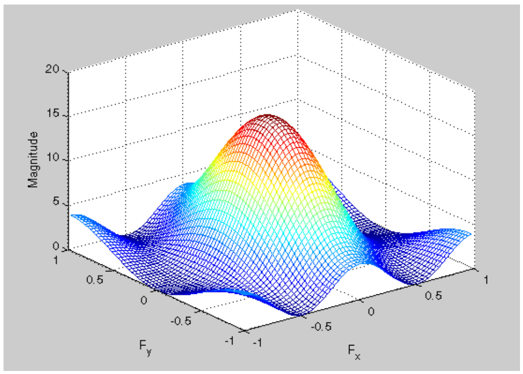


Figure 4: **Coefficients and Frequency Response for Filter Strength 3**

$$\text{Filter 4}=\begin{bmatrix}01210\\12221\\01210\end{bmatrix}$$

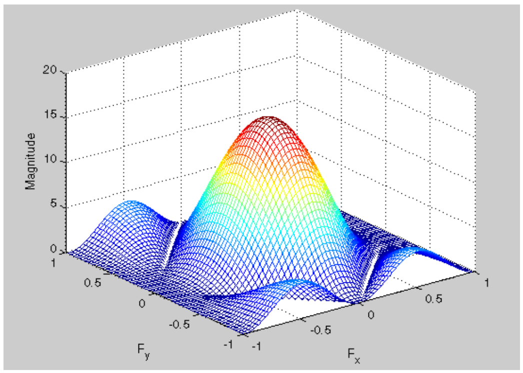


Figure 5: **Coefficients and Frequency Response for Filter Strength 4**

Processor Interfaces

The Image Noise Reduction core supports the following three processor interface options:

- EDK pCore Interface
- General Purpose Processor Interface
- Constant Interface

The processor interfaces provide the system designer with the ability to dynamically control the parameters within the core.

EDK pCore Interface

There are multiple imaging applications that include an embedded processor that can dynamically control the parameters within an integrated system. The CORE Generator software can be used to generate the direct pCore interface. The Xilinx MicroBlaze™ processor can be used to control directly the hardware added to an EDK project as a hardware peripheral. This pCore provides a memory-mapped interface for the programmable registers within the core, which are described in Table 1.

Table 1: **EDK pCore Interface Register Descriptions**

Address Offset (hex)	Register Name	Access Type	Default Value (hex)	Description	
BASEADDR + 0x800	noise_reg00_control	R/W	0x00000001	Bit 0	Software enable • 0 – Not enabled • 1 – Enabled
				Bit 1	Host processor write done semaphore • 0 – Host processor actively updating registers • 1 – Register update completed by host processor
BASEADDR + 0x804	noise_reg01_reset	R/W	0x00000000	Bit 0	Software reset • 0 – Not reset • 1 – Reset
BASEADDR + 0x808	noise_reg02_status	R	0x00000000	Bit 7	Timing locked • 1 - Indicates that the timing module of the core has locked on the input timing signals and is generating stable output timing signals
BASEADDR+ 0x80C	noise_reg03_filt_strength	R/W	from GUI	Strength of the smoothing filter Possible values are 0, 1, 2, 3, and 4 1= weakest, 4 = strongest (that is, more noise reduction); setting the filter strength to 0 will bypass the smoothing filter	

All of the registers are readable, enabling the MicroBlaze processor to verify writes or read current values contained within the registers. The default values of some of the registers are defined in the CORE Generator – Graphical User Interface.

This core supports an enable/disable function. When disabled, the normal operation of the hardware is halted by blocking the propagation of all video signals. This function is controlled by setting the Software Enable, bit 0 of `noise_reg00_control` register, to 0; the default value of Software Enable is 1 (enabled).

The in-system reset of the core is controlled by asserting `noise_reg01_reset` (bit 0), which returns the filter strength to its default value, specified through the Graphical User Interface when the core is instantiated. The core control signals and output are forced to 0 until the software reset bit is deasserted.

The `noise_reg03_filt_strength` register is double buffered in hardware to ensure no image tearing happens if the filter strength value is modified in the active area of a frame. This double buffering provides system control that is more flexible and easier to use by decoupling the register updates from the blanking period, allowing software a much larger window with which to update the parameter values. The updated value for the filter strength register is latched into the shadow register immediately after writing, while the actual filter strength used is stored in the working register.

Any reads of registers during operation return the values stored in the shadow registers. The rising edge of `vblank_in` triggers the values from the shadow registers to be copied to the working registers when bit 1 of `noise_reg00_control` is set to 1. This semaphore bit helps to prevent changing the filter strength mid-frame.

Figure 6 shows a software flow diagram for updating registers during the operation of the core.

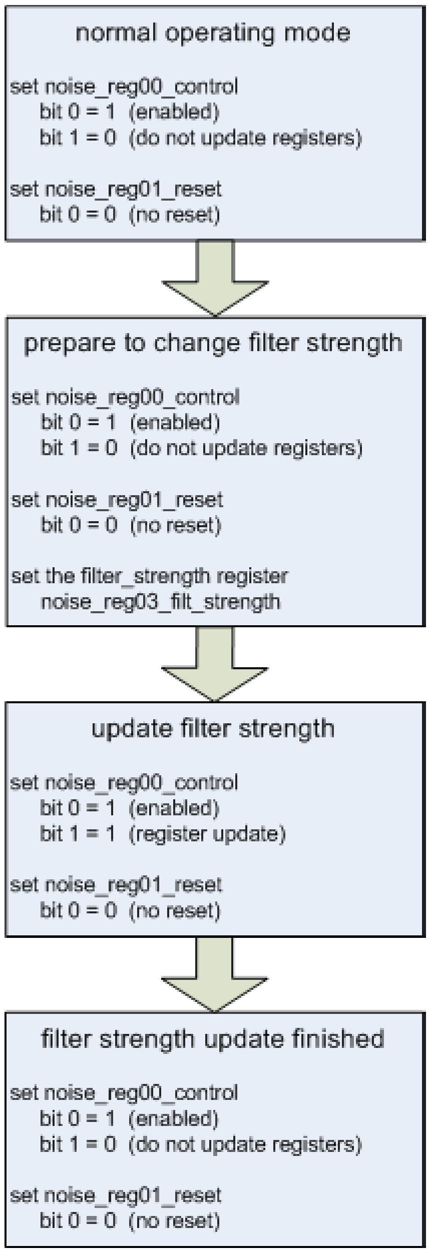


Figure 6: **Image Noise Reduction Programming Flow Chart**

See the EDK pCore Interface section of Core Symbol and Port Descriptions.

Programmer's Guide

The software API is provided to allow easy access to the Image Noise Reduction pCore's registers defined in Table 1. To utilize the API functions provided, the following two header files must be included in the user C code:

```
#include "noise.h"
#include "xparameters.h"
```

The hardware settings of your system, including the base address of your Image Noise Reduction core, are defined in the `xparameters.h` file. The `noise.h` file contains the macro function definitions for controlling the Image Noise Reduction pCore.

For examples on API function calls and integration into a user application, the drivers subdirectory of the pCore contains a file, `example.c`, in the `noise_v2_00_a/example` subfolder. This file is a sample C program that demonstrates how to use the Image Noise Reduction pCore API.

EDK pCore API Functions

This section describes the functions included in the C driver (`noise.c` and `noise.h`) generated for the EDK pCore API.

NOISE_Enable(uint32 BaseAddress);

- This macro enables an Image Noise Reduction instance.
- BaseAddress is the Xilinx EDK base address of the Noise Reduction core (from `xparameters.h`).

NOISE_Disable(uint32 BaseAddress);

- This macro disables an Noise Reduction instance.
- BaseAddress is the Xilinx EDK base address of the Noise Reduction core (from `xparameters.h`).

NOISE_Reset(uint32 BaseAddress);

- This macro resets an Noise Reduction instance. This reset effects the core immediately, and may cause image tearing. Reset affects the gain registers, forces `video_data_out` to 0, and forces timing signal outputs to their reset state until NOISE_ClearReset() is called.
- BaseAddress is the Xilinx EDK base address of the Noise Reduction core (from `xparameters.h`)

NOISE_ClearReset(uint32 BaseAddress);

- This macro clears the reset flag of the core, which allows it to re-sync with the input video stream and return to normal operation.
- BaseAddress is the Xilinx EDK base address of the Noise Reduction core (from `xparameters.h`).

Reading and Writing pCore Registers

Each software register defined in Table 1 has a constant defined in `noise.h` that is set to the offset for that register. Reading a value from a register uses the base address and offset for the register:

```
Xuint32 value = NOISE_ReadReg(XPAR_NOISE_0_BASEADDR, NOISE_REG03_FILT_STRENGTH);
```

This macro returns the 32-bit unsigned integer value of the register. The definition of this macro is:

NOISE_ReadReg(uint32 BaseAddress, uint32 RegOffset)

- Read the given register.
- BaseAddress is the Xilinx EDK base address of the Noise Reduction core (from `xparameters.h`).
- RegOffset is the register offset of the register (defined in Table 1).

To write to a register, use the NOISE_WriteReg() function using the base address of the Noise Reduction pCore instance (from `xparameters.h`), the offset of the desired register, and the data to write. For example:

```
NOISE_WriteReg(XPAR_NOISE_0_BASEADDR, NOISE_REG03_FILT_STRENGTH, 0);
```

The definition of this macro is:

NOISE_WriteReg(uint32 BaseAddress, uint32 RegOffset, uint32 Data)

- Write the given register.
- BaseAddress is the Xilinx EDK base address of the Noise Reduction core (from `xparameters.h`).
- RegOffset is the register offset of the register (defined in Table 1).
- Data is the 32-bit value to write to the register.

NOISE_RegUpdateEnable(uint32 BaseAddress);

- Calling RegUpdateEnable causes the Noise Reduction to start using the updated gain values on the next rising edge of `VBlank_in`. The user must manually disable the register update after a sufficient amount of time to prevent continuous updates.
- This function only works when the Noise Reduction core is enabled.
- BaseAddress is the Xilinx EDK base address of the Noise Reduction core (from `xparameters.h`)

NOISE_RegUpdateDisable(uint32 BaseAddress);

- Disabling the Register Update prevents the Noise Reduction gain registers from updating. Xilinx recommends that the Register Update be disabled while writing to the registers in the core, until the write operation is complete. While disabled, writes to the registers are stored, but do not affect the core's behavior.
- This function only works when the Noise Reduction core is enabled.
- BaseAddress is the Xilinx EDK base address of the Noise Reduction core (from `xparameters.h`)

General Purpose Processor Interface

The General Purpose Processor Interface exposes the filter strength register as a port. The General Purpose Processor Interface is provided as an option to design a system with a user-defined bus interface (decoding logic and register banks) to an arbitrary processor.

The filter strength port has the double-buffer control mechanism described in the previous section to prevent tearing. However, an external register (shadow register) has to be supplied by the user-defined bus interface. Values from this register bank (external to the Image Noise Reduction core) are copied over to the internal registers at the rising edge of `vblank_in` when bit 1 of the `noise_reg00_control` register is set to '1'.

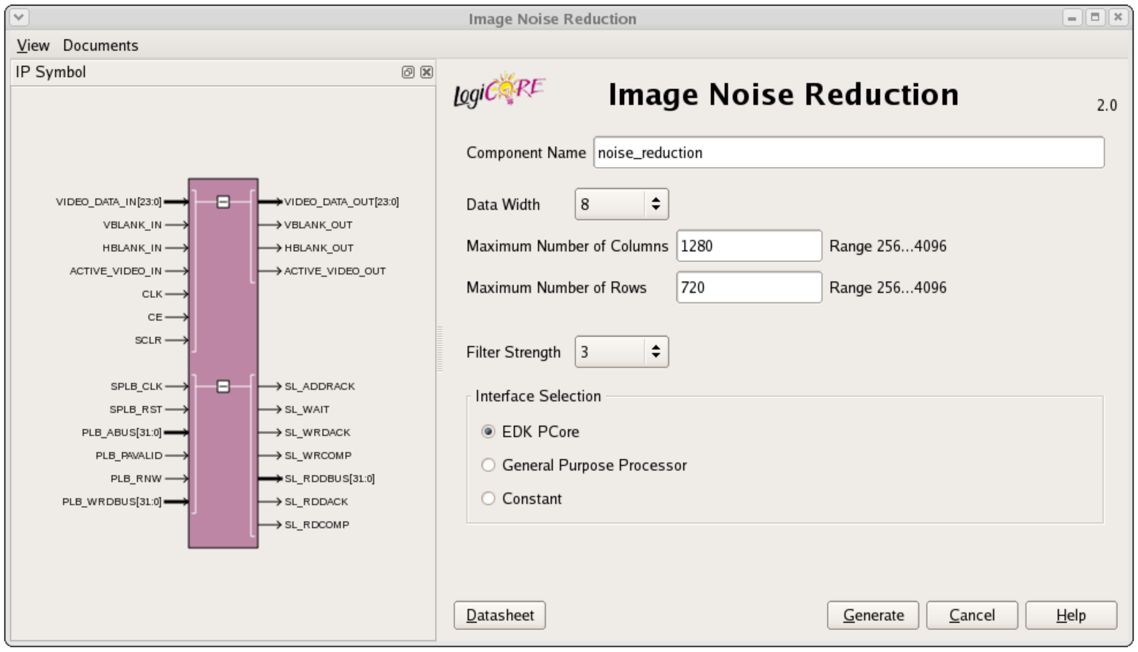
See the General Purpose Processor Interface section of Core Symbol and Port Descriptions.

Constant Interface

The Constant Interface does not provide an option for the filter strength to be changed in system. Also, there is no processor interface and the core is not programmable, but can be reset and enabled/disabled using the `SCLR` and `CE` ports. The ports for the Constant Interface are described in detail in the Constant Interface section of Core Symbol and Port Descriptions.

CORE Generator – Graphical User Interface

The Image Noise Reduction core is easily configured to meet the user's specific needs through the CORE Generator graphical user interface (GUI). This section provides a quick reference to the parameters that can be configured at generation time. Figure 7 shows the main Image Noise Reduction screen.



Component Name: noise_reduction

Data Width: 8

Maximum Number of Columns: 1280 Range 256...4096

Maximum Number of Rows: 720 Range 256...4096

Filter Strength: 3

Interface Selection
- [✓ EDK PCore]
- General Purpose Processor
- Constant

Figure 7: **Image Noise Reduction Main Screen**

The GUI displays a representation of the IP symbol on the left side, and the parameter assignments on the right side, which are described as follows:

- **Component Name:** The component name is used as the base name of output files generated for the module. Names must begin with a letter and must be composed from characters: a to z, 0 to 9 and "_".
- **Data Width (WIDTH):** Specifies the bit width of the input channel for each component. The allowed values are 8, 10, and 12.
- **Maximum Number of Columns (MAX_COLS):** Specifies the maximum number of columns that can be processed by the core. Permitted values are from 256 to 4096. Specifying this value is necessary to establish the internal widths of counters and control-logic components as well as the depth of line buffers. Using a tight upper-bound on possible values of `MAX_COLS` results in optimal block RAM usage. However, feeding the configured Image Noise Reduction instance timing signals that violate the `MAX_COLS` constraint leads to data and output timing signal corruption.
- **Maximum Number of Rows (MAX_ROWS):** Specifies the maximum number of rows that can be processed by the core. Permitted values are from 256 to 4096. Specifying this value is necessary to establish the internal widths of counters and control-logic components. Feeding the configured Image Noise Reduction instance timing signals that violate the `MAX_ROWS` constraint leads to data and output timing signal corruption.
- **Filter Strength:** Specifies which of the four smoothing filters to use. The allowed values are 0, 1, 2, 3, and 4. Filter Strength of 1 provides the weakest smoothing, and Filter Strength of 4 provides the strongest smoothing. Therefore, Filter Strength of 4 provides the most noise reduction. Setting the Filter Strength to 0 will bypass the smoothing filter.
- **Interface Selection:** As described in the previous sections, this option allows for the configuration of two different interfaces for the core.

- **EDK pCore Interface:** CORE Generator software generates a pCore that can be easily imported into an EDK project as a hardware peripheral, and filter strength can be programmed via a register. Double buffering is used to eliminate tearing of output images. See the EDK pCore Interface section of Processor Interfaces.
- **General Purpose Processor Interface:** CORE Generator software generates a set of ports to be used to program the core. See the Programmer's Guide section of Processor Interfaces.
- **Constant Interface:** The filter strength is constant, and therefore no programming is necessary. The constant value is set in the GUI.

Core Symbol and Port Descriptions

The Image Noise Reduction core can be configured with three different interface options, each resulting in a slightly different set of ports. The Image Noise Reduction core uses a set of signals that is common to all of the Xilinx Video IP cores called the Xilinx Streaming Video Interface (XSVI). The XSVI signals are shown in Figure 8 and described in Table 2.

Xilinx Streaming Video Interface

The Xilinx Streaming Video Interface (XSVI) is a set of signals common to all of the Xilinx video cores used to stream video data between IP cores. XSVI is also defined as an Embedded Development Kit (EDK) bus type so that the tool can automatically create input and output connections to the core. This definition is embedded in the pCORE interface provided with the IP, and it allows an easy way to cascade connections of Xilinx Video Cores. The Image Noise Reduction IP core uses the following subset of the XSVI signals:

- video_data
- vblank
- hblank
- active_video

Other XSVI signals on the XSVI input bus, such as video_clk, vsync, hsync, field_id, and active_chr do not affect the function of this core.

Note: These signals are neither propagated, nor driven on the XSVI output of this core.

The following is an example EDK Microprocessor Peripheral Definition (.MPD) file definition.

Input Side:

```
BUS_INTERFACE BUS = XSVI_NOISE_IN, BUS_TYPE = TARGET, BUS_STD = XSVI

PORT hblank_i       =hblank,      DIR=I,     BUS=XSVI_NOISE_IN

PORT vblank_i       =vblank,      DIR=I,     BUS=XSVI_NOISE_IN

PORT active_video_i =active_video,DIR=I,     BUS=XSVI_NOISE_IN

PORT video_data_i   =video_data,  DIR=I,VEC=[C_DATA_WIDTH-1:0], BUS=XSVI_NOISE_IN
```

Output Side:

```
BUS_INTERFACE BUS = XSVI_NOISE_OUT, BUS_TYPE = INITIATOR,  BUS_STD = XSVI
PORT hblank_o       =hblank,      DIR=I,                            BUS=XSVI_NOISE_OUT
PORT vblank_o       =vblank,      DIR=I,                            BUS=XSVI_NOISE_OUT
PORT active_video_o =active_video, DIR=I,                            BUS=XSVI_NOISE_OUT
PORT video_data_o  =video_data,   DIR=I, VEC=[3*C_DATA_WIDTH-1:0],BUS=XSVI_NOISE_OUT
```

The Image Noise Reduction IP core is fully synchronous to the core clock, clk. Consequently, the input XSVI bus is expected to be synchronous to the input clock, clk. Similarly, to avoid clock resampling issues, the output XSVI bus for this IP is synchronous to the core clock, clk. The video_clk signals of the input and output XSVI buses are not used.

Constant Interface

The Constant Interface has no ports other than the Xilinx Streaming Video Interface, clk, ce, and sclr signals, as this interface does not provide additional programmability. The Constant Interface Core Symbol is shown in Figure 8 and described in Table 2. The Xilinx Streaming Video Interface is a set of signals that is common in all interface options.

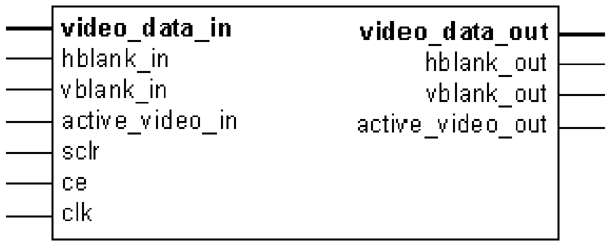


Figure 8: **Core Symbol for the Constant Interface**

Table 2: **Port Descriptions for the Constant Interface**

Port Name	Port Width	Direction	Description
video_data_in	*3*WIDTH*	*IN*	Data input bus
hblank_in	*1*	*IN*	Horizontal blanking input
vblank_in	*1*	*IN*	Vertical blanking input
active_video_in	*1*	*IN*	Active video signal input
video_data_out	*3*WIDTH*	*OUT*	Data output bus
hblank_out	*1*	*OUT*	Horizontal blanking output
vblank_out	*1*	*OUT*	Vertical blanking output
active_video_out	*1*	*OUT*	Active video signal output
clk	*1*	*IN*	Rising-edge clock
ce	*1*	*IN*	Clock enable (active high)
sclr	*1*	*IN*	Synchronous clear – reset (active high)

- **video_data_in:** This bus contains the luminance and chrominance inputs in the following order from MSB to LSB [Cb ; Cr : Y] or [U ; V ; Y]. Each component is expected in `WIDTH` bits wide unsigned integer representation.

Table 3:

Bits	***3WIDTH-1:2WIDTH***	***2WIDTH-1:WIDTH***	***WIDTH-1:0***
Video Data Signals	Cb or U	Cr or V	Y

- **hblank_in:** The `hblank_in` signal conveys information about the blank/non-blank regions of video scan lines.
- **vblank_in:** The `vblank_in` signal conveys information about the blank/non-blank regions of video frames, and is used by the Image Noise Reduction core to detect the end of a frame, when user registers can be copied to active registers to avoid visual tearing of the image.
- **active_video_in:** The `active_video_in` signal is high when valid data is presented at the input. Input data to the core, `video_data_in`, is ignored when `active_video_in` is low.
- **clk - clock:** Master clock in the design, synchronous with, or identical to, the video clock.
- **ce - clock enable:** Pulling `CE` low suspends all operations within the core. Outputs are held, and no input signals are sampled, except for reset (`SCLR` takes precedence over `CE`).
- **sclr - synchronous clear:** Pulling `SCLR` high results in resetting all output pins to zero or their default values. Internal registers within the XtremeDSP™ slice and D-flip-flops are cleared.
- **video_data_out:** This bus contains the luminance and chrominance outputs in the following order from MSB to LSB [Cb ; Cr : Y] or [U ; V ; Y]. Each component is expected in `WIDTH` bits wide unsigned integer representation.

Table 4:

Bits	***3WIDTH-1:2WIDTH***	***2WIDTH-1:WIDTH***	***WIDTH-1:0***
Video Data Signals	Cb or U	Cr or V	Y

- **hblank_out and vblank_out:** The corresponding input signals are delayed so blanking outputs are in phase with the video data output, maintaining the integrity of the video stream.
- **active_video_out:** The `active_video_out` signal is high when valid data is present at the output. When `active_video_out` is low, `video_data_out` is not valid even if it is non-zero.

EDK pCore Interface

The EDK pCore Interface generates Processor Local Bus (PLB4.6) interface ports in addition to the sclr and Xilinx Streaming Video Signals. The PLB bus signals are automatically connected when the generated pCore is inserted into an EDK project. The Core Symbol for the EDK pCore Interface is shown in Figure 9. The Xilinx Streaming Video Interface and sclr are described in the previous section (Table 2). For more information on the PLB bus signals, see Processor Local Bus (PLB) v4.6 [Ref 1].

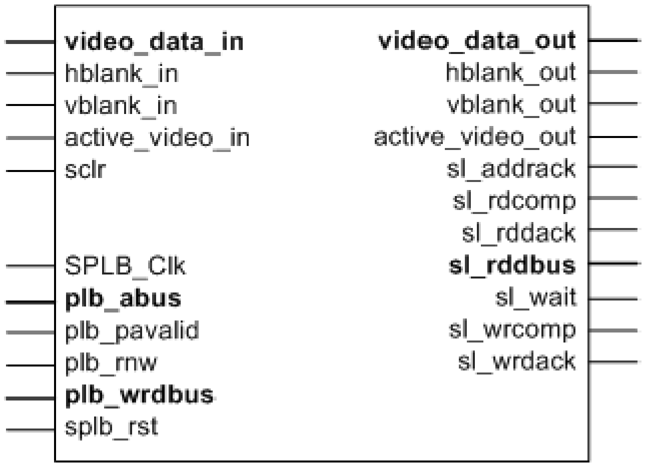


Figure 9: **Core Symbol for the EDK pCore Interface**

General Purpose Processor Interface

The General Purpose Processor Interface exposes the filter strength and control as ports. The Core Symbol for the General Purpose Processor Interface is shown in Figure 10. The Xilinx Streaming Video Interface is described in the previous section (Table 2). The ports are described in Table 5.

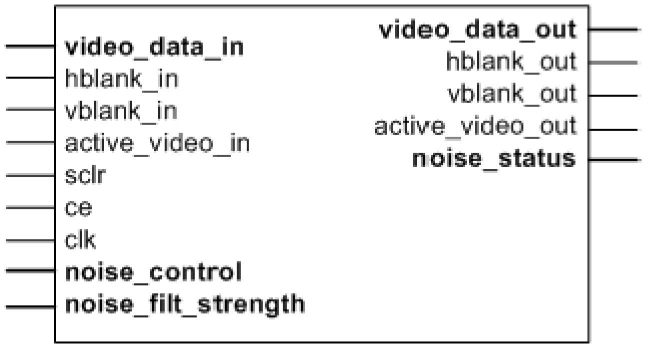


Figure 10: **Core Symbol for the General Purpose Processor Interface**

Table 5: **Optional Ports for the General Purpose Processor Interface**

Port Name	Port Width	Direction	Description
noise_control	*2*	*IN*	Bit 0: Software enable Bit 1: Host processor write done semaphore • 0 indicates host processor actively updating registers • 1 indicates register update completed by host processor
noise_filt_strength	*2*	*IN*	Strength of the smoothing filter Possible values are 0, 1, 2, 3, and 4 Select one of four smoothing filters with 1 being the weakest smoothing filter and 4 being the strongest smoothing filter; selecting 0 will bypass the smoothing filters
noise_status	*16*	*OUT*	Bit 7: Timing locked '1' indicates that the timing module of the core has locked on the input timing signals and is generating stable output timing signals

Control Signals and Timing

The propagation delay of the Image Noise Reduction core is one full scan line and 18 video clock cycles. The output timing signals (`vblank_out`, `hblank_out` and `active_video_out`) are delayed appropriately so that the output video data is framed correctly by the timing signals.

Deasserting `CE` suspends processing, which may be useful for data-throttling, to temporarily cease processing of a video stream to match the delay of other processing components.

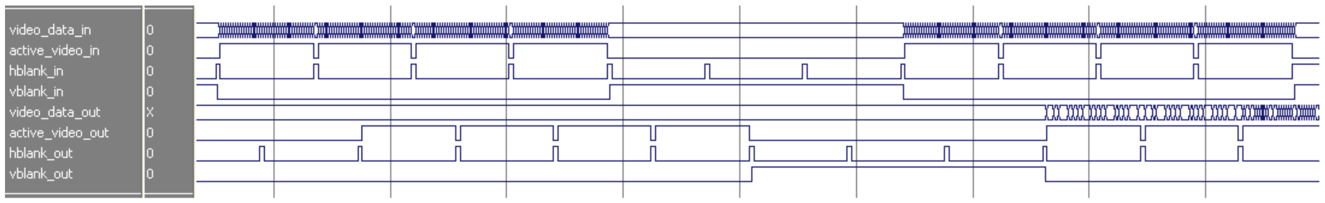


Figure 11: **Timing Example**

When `SCLR` is asserted, all data and control signal outputs are forced to zero. If the input control signal was high at the time `SCLR` was asserted, the corresponding output control signal goes low and stays low until the next expected rising edge.

The control signals `vblank_out`, `hblank_out`, and `active_video_out` are created using a timing detector and generator within the core. The internal timing module assumes the following:

- One horizontal blanking period per row
- One vertical blanking period per frame
- A minimum active frame size of four rows and eight columns
- A minimum horizontal blanking period of two columns
- A minimum vertical blanking period of three rows

During the detection of the timing control signals, the core cannot guarantee the correct video data output. Therefore, the data output, `video_data_out`, of the first frame of data is set to zero even though `active_video_out` is high.

Core Resource Utilization and Performance

Information presented in Table 6, Table 7, Table 8, and Table 9 is a guideline to the resource utilization of the Image Noise Reduction core for Spartan-6, Virtex-6, Virtex-5 and Spartan-3ADSP FPGA families. This core does not use any dedicated I/O or clock resources. The design was tested using Xilinx ISE® software v12.4 tools with area constraints and default tool options.

For an accurate measure of the usage of device resources (for example, block RAMs, flip-flops, and LUTs) for a particular instance, click **View Resource Utilization** in the CORE Generator interface after generating the core.

Table 6: **Resource Utilization and Target Speed for Spartan-6 - xc6slx150-2fgg676**

Data Width	Maximum Number of Columns and Rows	FFs	LUTs	Slices	Block RAMs	DSP48A1s	Clock Frequency (MHz)
8	1024	1505	1210	470	3(16K)	3	161
8	2200	1567	1307	503	12(16K)	3	159
10	1024	1787	1410	536	3(16K)+1(8K)	3	161
10	2200	1851	1523	551	15(16K)	3	164
12	1024	2071	1586	643	4(16K)	3	167
12	2200	2137	1730	641	18(16K)	3	167

1. Speedfile: PRODUCTION 1.13b 2010-11-15

Table 7: **Resource Utilization and Target Speed for Virtex-6 - xc6vsx315t-1ff1156**

Data Width	Maximum Number of Columns and Rows	FFs	LUTs	Slices	Block RAMs	DSP48E1s	Clock Frequency (MHz)
8	1024	1453	1226	469	1(36K)+1(18K)	3	250
8	2200	1517	1306	394	5(36K)+1(18K)	3	334
10	1024	1723	1409	475	2(36K)	3	266
10	2200	1787	1493	486	6(36K)+2(18K)	3	246
12	1024	1993	1622	494	2(36K)	3	244
12	2200	2057	1686	531	8(36K)	3	281

1. Speedfile: PRODUCTION 1.11d 2010-11-15

Table 8: **Resource Utilization and Target Speed for Spartan-3A DSP - xc3sd3400a-5fg676**

Data Width	Maximum Number of Columns and Rows	FFs	LUTs	Slices	Block RAMs	DSP48As	Clock Frequency (MHz)
8	1024	1470	1352	1095	3	3	154
8	2200	1514	1394	1220	12	3	153
10	1024	1740	1599	1314	4	3	153
10	2200	1784	1641	1353	15	3	153
12	1024	2010	1881	1573	4	3	153
12	2200	2054	1922	1465	18	3	152

1. Speedfile: PRODUCTION 1.33 2010-11-15

Table 9: **Resource Utilization and Target Speed for Virtex-5 - xc5vsx50t-1ff665**

Data Width	Maximum Number of Columns and Rows	FFs	LUTs	Slices	Block RAMs	DSP48Es	Clock Frequency (MHz)
8	1024	1450	1241	543	1(36K)+1(18K)	3	245
8	2200	1514	1314	539	5(36K)+1(18K)	3	232
10	1024	1720	1453	558	2(36K)	3	266
10	2200	1784	1521	683	6(36K)+2(18K)	3	228
12	1024	1990	1691	727	2(36K)	3	229
12	2200	2054	1758	734	8(36K)	3	246

1. Speedfile: PRODUCTION 1.72 2010-11-15

Known Issues

For for the latest Known Issues see XTP025.

References

1. Processor Local Bus (PLB) v4.6

Support

Xilinx provides technical support for this LogiCORE product when used as described in the product documentation. Xilinx cannot guarantee timing, functionality, or support of product if implemented in devices that are not defined in the documentation, if customized beyond that allowed in the product documentation, or if changes are made to any section of the design labeled *DO NOT MODIFY*.

License Options

The Image Noise Reduction core provides the following three licensing options:

- Simulation Only
- Full System Hardware Evaluation
- Full

After installing the required Xilinx ISE software and IP Service Packs, choose a license option.

Simulation Only

The Simulation Only Evaluation license key is provided with the Xilinx CORE Generator tool. This key lets you assess core functionality with either the example design provided with the Image Noise Reduction core, or alongside your own design and demonstrates the various interfaces to the core in simulation. (Functional simulation is supported by a dynamically generated HDL structural model.)

No action is required to obtain the Simulation Only Evaluation license key; it is provided by default with the Xilinx CORE Generator software.

Full System Hardware Evaluation

The Full System Hardware Evaluation license is available at no cost and lets you fully integrate the core into an FPGA design, place-and-route the design, evaluate timing, and perform functional simulation of the Image Noise Reduction core using the example design and demonstration test bench provided with the core.

In addition, the license key lets you generate a bitstream from the placed and routed design, which can then be downloaded to a supported device and tested in hardware. The core can be tested in the target device for a limited time before timing out (ceasing to function), at which time it can be reactivated by reconfiguring the device.

To obtain a Full System Hardware Evaluation license, do the following:

1. Navigate to the product page for this core.
2. Click Evaluate.
3. Follow the instructions to install the required Xilinx ISE software and IP Service Packs.

Full

The Full license key is available when you purchase the core and provides full access to all core functionality both in simulation and in hardware, including:

- Functional simulation support
- Full implementation support including place and route and bitstream generation
- Full functionality in the programmed device with no time outs

To obtain a Full license key, you must purchase a license for the core. Click on the "Order" link on the Xilinx.com IP core product page for information on purchasing a license for this core. After doing so, click the "How do I generate a license key to activate this core?" link on the Xilinx.com IP core product page for further instructions.

Installing Your License File

The Simulation Only Evaluation license key is provided with the ISE CORE Generator system and does not require installation of an additional license file. For the Full System Hardware Evaluation license and the Full license, an email will be sent to you containing instructions for installing your license file. Additional details about IP license key installation can be found in the ISE Design Suite Installation, Licensing and Release Notes document.

Revision History

The following table shows the revision history for this document:

Date	Version	Description of Revisions
12/02/09	1.0	Initial Xilinx release.
07/23/10	1.1	Fixed CR 54061 by adding Xilinx Streaming Video Interface (XSVI) information.
12/14/10	2.0	Updated for core version 2.0.

Notice of Disclaimer