

Key Design Features

- Synthesizable, technology independent IP Core for FPGA, ASIC and SoC
- Supplied as human readable VHDL (or Verilog) source code
- Video interlacer converts any progressive video format to its interlaced equivalent – e.g. 1080p to 1080i
- Supports 24-bit RGB or 4:4:4 YCbCr pixels
- Supports all video resolutions up to $2^{16} \times 2^{16}$
- Fully pipelined architecture with simple valid-ready flow control
- Self-flushing design operates like a simple FIFO
- One frame in generates one interlaced field out
- Output rate is one pixel per clock
- Supports 300 MHz+ operation on basic FPGA devices

Applications

- Conversion of all standard and custom video formats such as 1920x1080p to 1920x1080i, 720x480p to 720x480i etc.
- Video solutions for flat panel displays, portable devices, video consoles, video format converters, set-top boxes, digital TV etc.

Pin-out Description

| Pin name | I/O | Description | Active state |
|------------------------|-----|---|-----------------|
| clk | in | Synchronous clock | rising edge |
| reset | in | Asynchronous reset | low |
| pixels_per_line [15:0] | in | Number of pixels per input line of video | data |
| lines_per_frame [15:0] | in | Number of lines per input frame of video | data |
| pixin [23:0] | in | 24-bit input pixel | data |
| pixin_vsync | in | Vertical sync in | high |
| pixin_hsync | in | Horizontal sync in | high |
| pixin_val | in | Input pixel valid | high |
| pixin_rdy | out | Ready to accept input pixel (handshake signal) | high |
| pixout [23:0] | out | 24-bit output pixel | data |
| pixout_field | out | Output field number | 0: odd, 1: even |
| pixout_vsync | out | Vertical sync out | high |
| pixout_hsync | out | Horizontal sync out | high |
| pixout_val | out | Output pixel valid | high |
| pixout_rdy | in | Ready to accept output pixel (handshake signal) | high |

Block Diagram

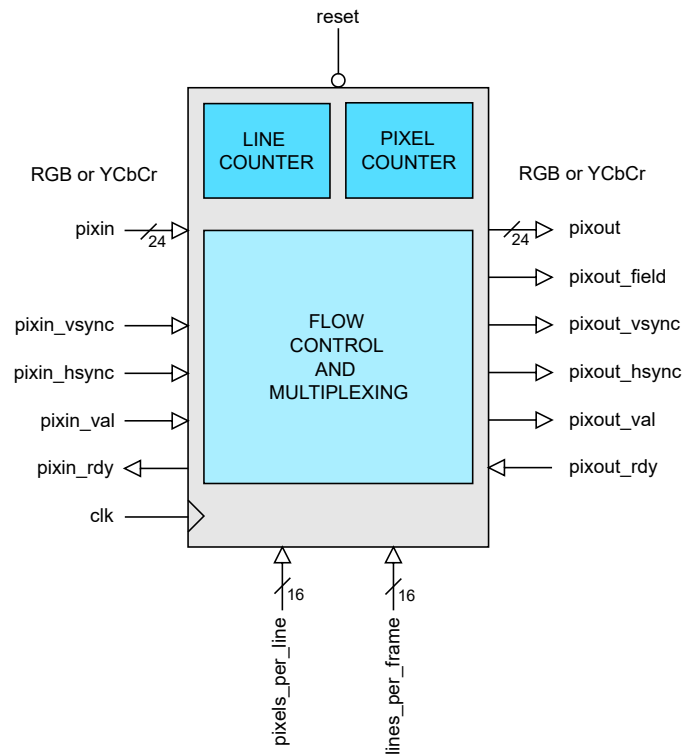


Figure 1: Video interlacer architecture

General Description

The INTERLACER IP Core (Figure 1) is a fully pipelined video interlacer solution that converts any progressive video format into its interlaced equivalent. The format of the input video is defined by the parameters *pixels_per_line* and *lines_per_frame*. These values specify the size of one input frame of video in pixels and lines. Each interlaced output field will have half the number of lines as an input frame.

The input and output interfaces are streaming interfaces that follow a simple valid-ready pipeline protocol¹. Input pixels and syncs are sampled on the rising edge of *clk* when *pixin_val* and *pixin_rdy* are both high. Likewise, output pixels and syncs are sampled on the rising edge of *clk* when *pixout_val* and *pixout_rdy* are high. The interfaces are compatible with all Zipcores video IP Cores and allow for easy connectivity between modules.

The input sync signals *vsync* and *hsync* are sideband flags that are coincident with the first pixel of a frame and the first pixel of a line respectively. The output sync signals are coincident with the first pixel of an output *field*. Note that the output interface has an additional *field* flag that identifies whether the field is odd or even. This field flag is held high or low for the duration of the output field.

Note that if no flow control is required in the design and the output is guaranteed to accept pixels without stalling, then the signal *pixout_rdy* may be tied high and the signal *pixin_rdy* may be ignored.

¹ Please see Zipcores application note: [app_note_zc001.pdf](#) for more examples of how to use the valid-ready pipeline protocol

Functional Timing

Figure 2 shows the signalling at the input to the interlacer at the start of a new frame. The first line of a new frame begins with *pixin_vsync* and *pixin_hsync* asserted high together with the first pixel. Note that the signals *pixin*, *pixin_vsync* and *pixin_hsync* are only valid if *pixin_val* is also asserted high.

In addition, the diagram shows what happens when *pixin_rdy* is de-asserted. In this case, the pipeline is stalled and the upstream interface must hold-off before further pixels are processed.

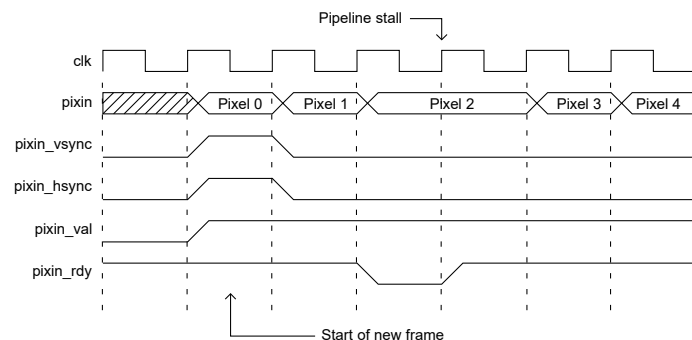


Figure 2: First pixel of a new input frame (and line) – also showing an example of a pipeline stall for one clock cycle

Figure 3 shows the signalling at the beginning of a new line only. The first pixel of a new line is specified with *pixin_vsync* asserted low and *pixin_hsync* asserted high. This time, there is no pipeline stalling shown.

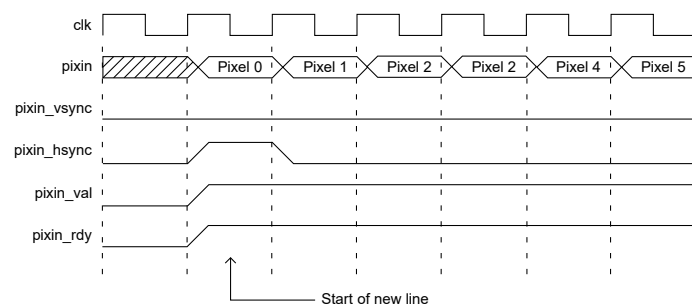


Figure 3: First pixel of a new input line

Finally, figure 4 shows the signalling at the output of the interlacer. The output uses exactly the same protocol as the input with the exception of the additional *pixout_field* flag. The *pixout_val* flag indicates whether the output field is odd or even.

In this particular example, it shows *pixout_val* de-asserted for 1 clock-cycle, in which case, the output pixel should be ignored. Remember that transfers at the interface are only permitted when valid and ready are both simultaneously high.

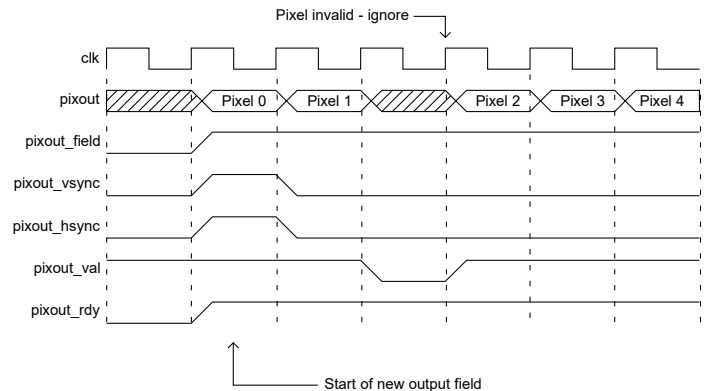


Figure 4: First pixel of a new output field – also showing invalid output pixel for one clock cycle

Source File Description

All source files are provided as text files coded in VHDL. The following table gives a brief description of each file.

| Source file | Description |
|-----------------------|------------------------------------|
| video_in.txt | Text-based source video file |
| video_file_reader.vhd | Reads text-based source video file |
| interlacer.vhd | Interlacer top-level component |
| interlacer_bench.vhd | Top-level test bench |

Functional Testing

An example VHDL testbench is provided for use in a suitable VHDL simulator. The compilation order of the source code is as follows:

1. video_file_reader.vhd
2. interlacer.vhd
3. interlacer_bench.vhd

The VHDL testbench instantiates the INTERLACER component with the parameters set up for a 720 x 480 source image.

The source video for the simulation is generated by the video file-reader component. This component reads a text-based file which contains the RGB pixel data. The text file is called *video_in.txt* and should be placed in the top-level simulation directory.

The file *video_in.txt* follows a simple format which defines the state of signals: *pixin_val*, *pixin_vsync*, *pixin_hsync* and *pixin* on a clock-by-clock basis. An example file might be the following:

```

1 1 1 00 11 22 # pixel 0 line 0 (start of frame)
1 0 0 33 44 55 # pixel 1
0 0 0 00 00 00 # don't care!
1 0 0 66 77 88 # pixel 2
.
1 0 1 00 11 22 # pixel 0 line 1 etc..

```

In this example, the first line of the *video_in.txt* file asserts the input signals *pixin_val* = 1, *pixin_vsync* = 1, *pixin_hsync* = 1 and *pixin* = 0x001122.

The simulation must be run for at least 20 ms during which time an output text file called *video_out.txt* will be generated. This file contains a sequential list of 24-bit output pixels in the same format as *video_in.txt*.

The example provided generates interlaced video from a 720 x 480 input source. Figure 5 and 6 show the resulting odd and even output fields which are each 720 x 240 pixels in resolution.

```

ames          384 fi
192 frames
96 frames
64 frames
32 frames

```

Figure 5: Simulation output – odd field (0)

```

ames          384 fi
192 frames
96 frames
64 frames
32 frames

```

Figure 6: Simulation output – even field (1)

Revision History

| Revision | Change description | Date |
|----------|--|------------|
| 1.0 | Initial revision | 12/06/2016 |
| 1.1 | Added timing waveforms to datasheet. Updated synthesis results in line with minor functional changes to the source code. | 01/03/2019 |
| | | |
| | | |
| | | |

Synthesis and Implementation

Only one file is required for synthesis. This is the top-level source file: 'interlacer.vhd'.

The VHDL core is designed to be technology independent. However, as a benchmark, synthesis results have been provided for the Xilinx® 7-series FPGAs. Synthesis results for other FPGAs and technologies can be provided on request.

The speed of the design may be improved by tying the signal *pixout_rdy* low. This may be possible if the designer knows that the pipeline downstream of the interlacer IP core will always be able to accept output pixels.

The table below shows resource usage after place and route of the design.

XILINX® 7-SERIES FPGAS

| Resource type | Artix-7 | Kintex-7 | Virtex-7 |
|----------------------|---------|----------|----------|
| Slice Register | 65 | 65 | 65 |
| Slice LUTs | 87 | 87 | 87 |
| Block RAM | 0 | 0 | 0 |
| DSP48 | 0 | 0 | 0 |
| Occupied Slices | 28 | 25 | 22 |
| Clock freq. (approx) | 300 MHz | 400 MHz | 500 MHz |

Mouser Electronics

Authorized Distributor

Click to View Pricing, Inventory, Delivery & Lifecycle Information:

[Zipcores:](#)

[SKU97](#)