HD_SMPTE_DECODER

High-Definition SMPTE Decoder with Colour-Space Converter
Rev. 1.1

Key Design Features

- Synthesizable, technology independent IP Core for FPGA, ASIC and SoC
- Supplied as human-readable VHDL (or Verilog) source code
- Converts industry standard 20-bit 4:2:2 SMTPE digital video to 24-bit RGB pixels + sync flags
- Integrated 4:2:2 YCbCr to RGB888 colour-space converter
- All 720p and 1080p (progressive) video formats supported including SMPTE 296M and SMPTE 274M
- All signals synchronous with the pixel clock
- Small implementation size ideal for all types of FPGA
- Compatible with a wide range of video decoder ICs
- Full HD1080p60 video supported on most devices[1]

Applications

- Conversion of progressive SMPTE video formats
- Connectivity with a wide range of commercially available video decoder ICs, 3G-SDI, HD-SDI and HDMI receivers
- Simple and cost-effective method for capturing digital video into your FPGA, ASICor SoC

Generic Parameters

Generic name	*Description*	*Type*	*Valid range*
hd_mode	Input video mode	integer	0: 720p 1: 1080p

Pin-out Description

Pin name	*I/O*	*Description*	*Active state*
clk	in	Pixel clock	rising edge
reset	in	Asynchronous reset	low
video_y [9:0]	in	Y channel input	data
video_c [9:0]	in	CbCr channel input	data
pixout [23:0]	out	24-bit RGB888 pixel out	data
pixout_vsync	out	Vertical sync out (Coincident with the first pixel of a frame)	high
pixout_hsync	out	Horizontal sync out (Coincident with the first pixel of a line)	high
pixout_val	out	Output pixel valid	high

Block Diagram

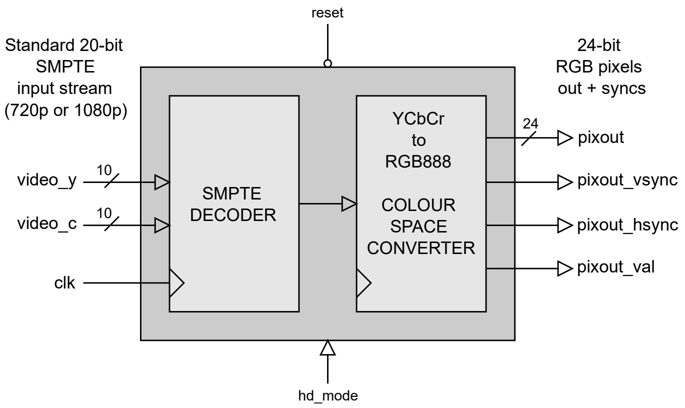

Figure 1: SMPTE Decoder simplified architecture

General Description

The HD_SMPTE_DECODER IP Core (Figure 1) is a digital video decoder with integrated colour-space converter. It's function is to extract the valid pixels from a standard SMPTE video stream and convert them to 24-bit RGB pixels for subsequent processing.

Video decoding begins after reset is de-asserted and on the rising-edge of *clk.* Output RGB pixels are generated on each active line starting with the first frame.

4:2:2 Pixels are extracted from the SMPTE input stream and converted to RGB888 format. These pixels are then presented at the output of the decoder together with the sync flags. The *pixout_val* signal is active high for the duration of each active line and it is disabled during blanking periods. All signals are synchronous with the pixel clock.

SMPTE Decoder

The decoder samples the incoming SMPTE input and looks for the first active line at the start of a new frame. Once this is detected, output pixels are generated on a line-by-line basis. After a system reset, the decoder will revert to it's initial state and stop generating output pixels. Decoding will then begin again with the first active line of the next frame.

When the generic parameter *hd_mode* is set to '0', the decoder expects a 720p format video stream with 720 active lines and 1280 pixels per line. Conversely, when *hd_mode* is set to '1' then 1080 lines of 1920 pixels per line is expected. The decoder supports all *progressive* video refresh rates as defined in the SMPTE 296M and SMPTE 274M specifications[2].

Note that during blanking periods, no valid output pixels are generated. Decoding is only concerned with extraction of active pixels from the SMPTE stream. Valid 4:2:2 YCbCr pixels are passed to the Colour-Space Converter module where they are converted to 24-bit RGB.

[1] Xilinx® 7-series FPGAs used as a benchmark

[2] Auto detect of the input video mode is an option. Interlaced modes may also supported on request. Please contact Zipcores for more information.

Colour-space converter

Chroma values from the input pixels (Cb, Cr) are duplicated every second pixel and then converted to the RGB888 colour-space using the following formulas:

$$R = 1.164(Y - 16) + 1.596(Cr - 128)$$
$$G = 1.164(Y - 16) + 0.813(Cr - 128) - 0.391(Cb - 128)$$
$$B = 1.164(Y - 16) + 2.018(Cb - 128)$$

In addition, the colour-space converter also generates correctly aligned vsync, hsync and valid flags. All output flags are qualified by the *pixout_val* signal and are synchronous with the pixel clock. Example timing waveforms are given in the functional timing section below.

Functional Timing

Figure 2 shows the typical format of an SMPTE video stream into the decoder. The 10-bit Y/C values are sampled in parallel on a rising clock-edge when the system reset is disabled.

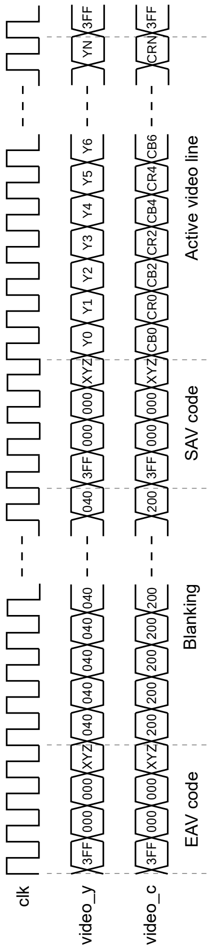

Figure 2: Typical SMPTE 296M/274M input stream

Example output decoder waveforms are shown in Figure 3. Output pixels and syncs are transferred on a rising clock-edge when *pixout_val* is high. The signal *pixout_val* is low during blanking periods.

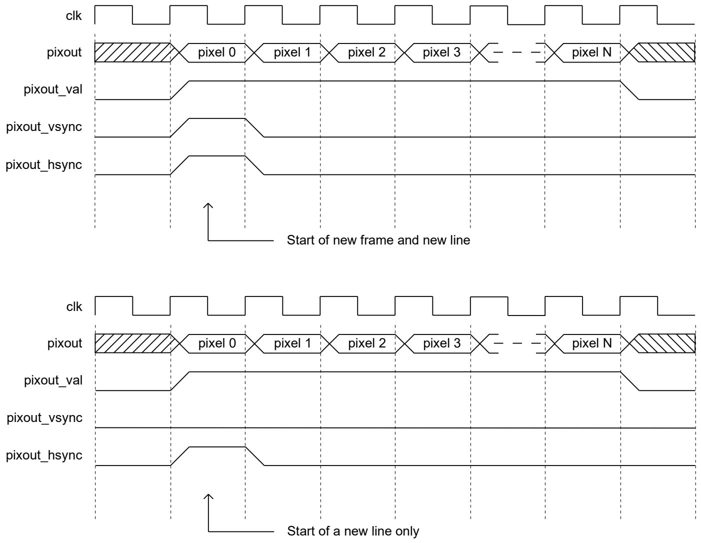

Figure 3: Output waveforms showing the start of a new frame and the start of a new line

Source File Description

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

Source file	***Description***
smpte_file_reader.vhd	SMPTE text file reader
hd_smpte_dec.vhd	Main decoder component
hd_smpte_dec_csc.vhd	Colour-space converter
hd_smpte_decoder.vhd	Top-level component
hd_smpte_decoder_bench.vhd	Top-level test bench

Functional Testing

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

1. hd_smpte_dec_csc.vhd
2. hd_smpte_dec.vhd
3. hd_smpte_decoder.vhd
4. hd_smpte_decoder_bench.vhd
5. hd_smpte_file_reader.vhd

The VHDL test bench instantiates the HD_SMPTE_DECODER IP Core component with the video format set to either 720p or 1080p. The source video for the simulation is generated by the text file reader component. This component reads a text-based file which contains the SMPTE encoded data with each Y/C pixel on a separate line. The text file is called *smpte_in.txt* and should be placed in the top-level simulation directory.

The simulation must be run for at least 50 ms during which time all the outputs are captured to a text file called *video_out.txt*. This file contains a sequential list of output pixels and flags which may be processed and used to generate an output image in software[3].

Figure 4 below shows the results of simulation after decoding a 720p SMPTE input stream. Original full resolution bitmap images are available on request.

Figure 4: Video output frame from the example simulation

Synthesis and Implementation

The files required for synthesis and the design hierarchy is shown below:

- hd_smpte_decoder.vhd
  - hd_smpte_dec.vhd
  - hd_smpte_dec_csc.vhd

The IP 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.

If the *video_y* and *video_c* input signals enter the device externally then we recommend placing these signals in the input I/O or pads of the device. They should also be placed locally together to avoid unnecessary skew between individual data bits. Other than this, there are no special constraints required for synthesis. The IP Core is completely technology independent.

Trial synthesis results are shown with the generic *hd_mode* parameter set to '0' for 720p video. The resource usage is specified after Place and Route.

XILINX® 7-SERIES FPGAS

Resource type	***Artix-7***	***Kintex-7***	***Virtex-7***
Slice Register	160	160	160
Slice LUTs	92	91	90
Block RAM	0	0	0
DSP48	12	12	12
Occupied Slices	61	65	62
Clock freq. (approx)	250 MHz	300 MHz	350 MHz

Revision History

Revision	***Change description***	***Date***
1.0	Initial revision	22/02/2018
1.1	First official release	16/10/2019

[3] We can provide a number of scripts for generating SMPTE input streams and parsing the output results of simulation. Please contact Zipcores for details.