ALSE Video Reference Designs using AVDB

The FPGA & HDL Design Experts
www.ALSE-FR.com

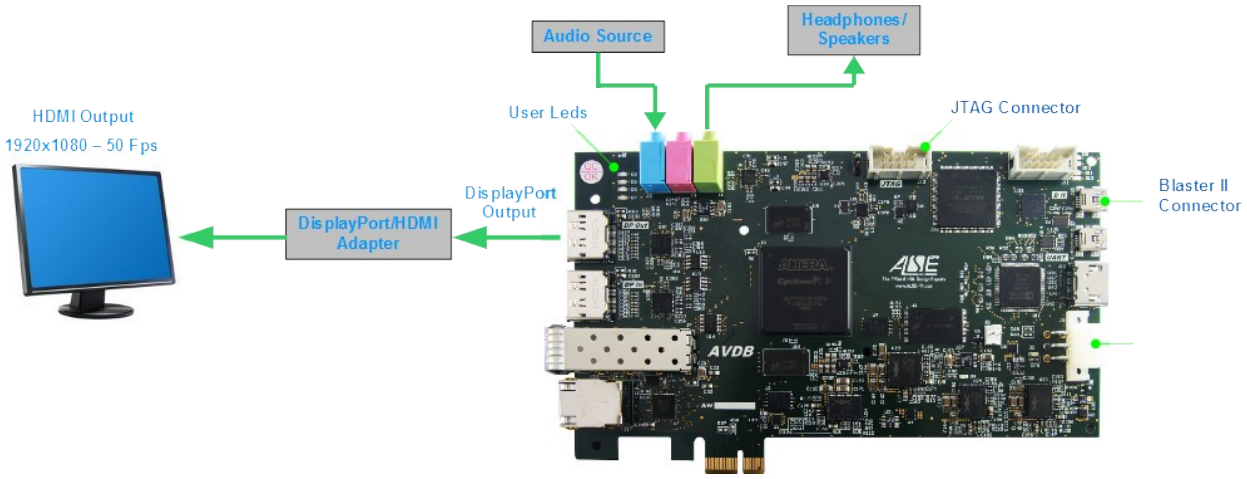
These designs illustrate the use of AVDB to implement various Video and Audio applications.

Design #1 : HDMI Output

This simple design example shows how to generate a simple video project using Qsys and test it on AVDB : the Advanced Video Development Board from ALSE.

This design uses the HDMI Output IP developed by ALSE.

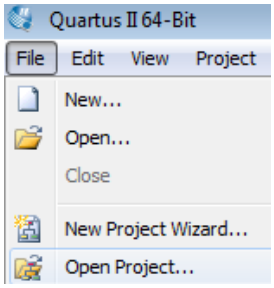
Hardware setup

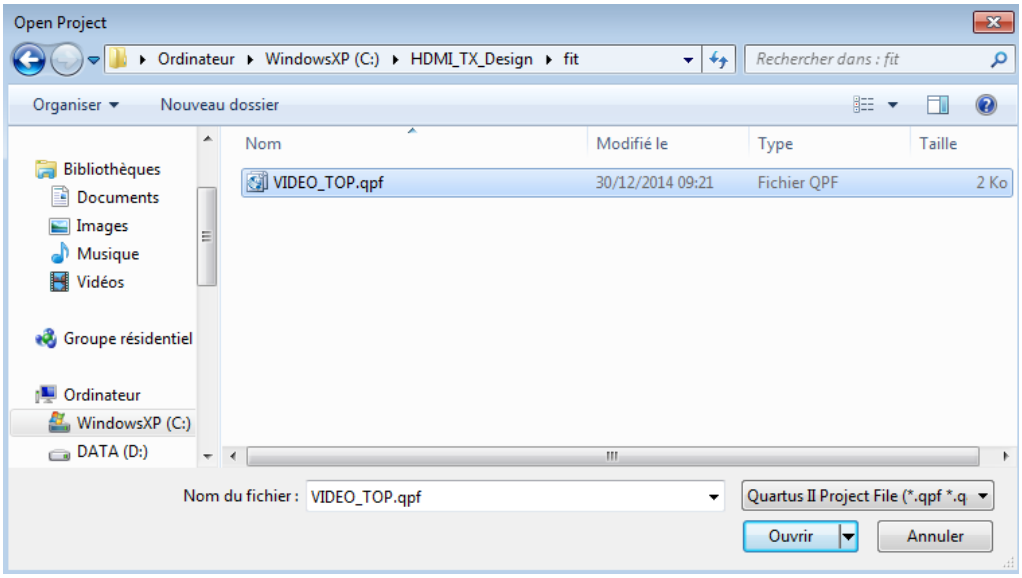


Note : Though we will connect to the DP (Display Port) connector on AVDB, the actual logical signals going through this port will be *HDMI* signals. The Physical levels are then shifted to HDMI standard by a small (active) “DP → HDMI conversion cable”.

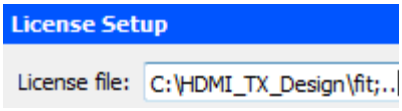
Step-by-Step Instructions

✔ Launch Quartus II and open the project named “VIDEO_TOP.qpf” located in the HDMI_TX_Design directory :





✔ Once the Quartus II Project is opened set up the license file to use to compile this design.
Under Quartus from the menu select **Tools > License setup...**
In the License File area, **add the directory of the Quartus Project** where the license file “alse_opencore_xxx.ocp” is located.



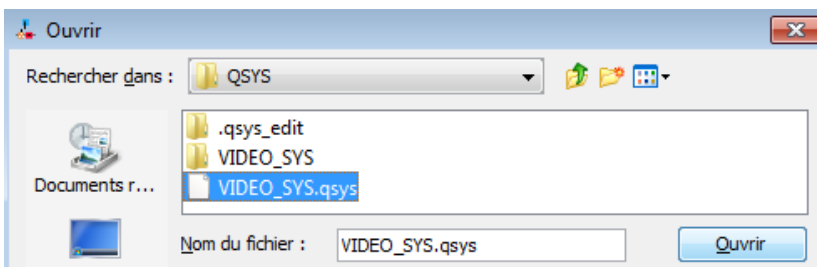
✔ When the new license file location has been added to the current license file a new Licensed function should be visible in the list :

Vendor	Product	Version
ALSE (7D3A)	FFFF	2015.01

✔ Click OK to close this window.

✔ Under Quartus II **launch Qsys** by clicking on the following icon (or from the menu Tools → Qsys)

✔ In Qsys open the component named “**VIDEO_SYS.qsys**” located in “**../src/qsys/**”.
At the question : “Save changes to unnamed ?” → select “**Don't save**”.



Once opened, the system should look like this :

Use	C...	Name	Description	Export	Clock
☑		⊞ SYS_CLK	Clock Source		*exported*
☑		⊞ BOARD_CONFIG	AVDB Configuration Block		SYS_CLK
☑		⊞ VIDEO_CLK	Clock Source		*exported*
☑		⊞ VIDEO_GEN	ALSE Video Pattern Generator		VIDEO_CLK
☑		⊞ AUDIO_GEN	ALSE Audio Generation Block		SYS_CLK
☑		⊞ HDMI_OUT	ALSE HDMI TX IP		*multiple*

It includes :

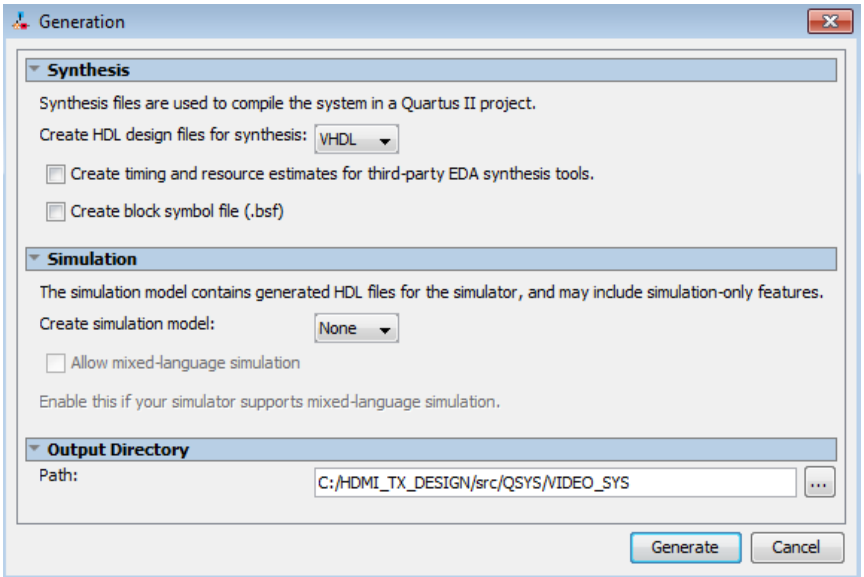
- A **clock source** component for the system clock signal (100 MHz generated by a PLL at Top level)
- The **AVDB Configuration Block.**
  This component has been created by ALSE to configure the devices connected to the FPGA :
  – The External Programmable Clock Generator (SiLabs PLL) in order to generate a clock running at 148.5 MHz, required by the Transceiver(s) for the HDMI Output.
  – The Re-driver of the DisplayPort Interface
  – The on-board Audio Codec

  Note : The configuration of these peripherals could be adjusted at run-time through a serial port.
- A **video clock source** for the Video clock signal (we will use a 148.5 MHz Pixel Clock)
- An **Audio Generation** Block created by ALSE to convert an I2S audio interface into an Avalon-ST Streaming interface composed of ancillary packets (not enabled at this time, we will use it later).
- The **HDMI out** IP from ALSE.
- And several other components...

Generating the System

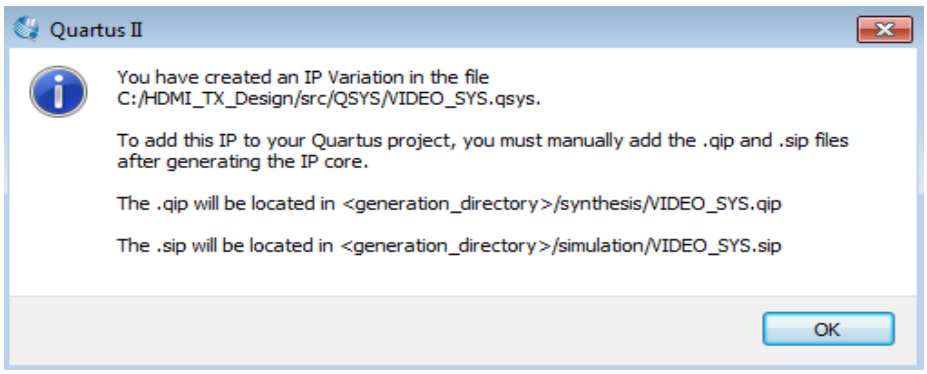
✔ From the **Generate** menu click on **Generate HDL**...

✔ In the ***Generation*** Pop-up window, configure the generation as follows :



✔ Once generated **close Qsys** and go back to the Quartus II Software

✔ Click **OK** if the following message appears :



✔ **Launch the compilation** of the design by clicking on the icon 

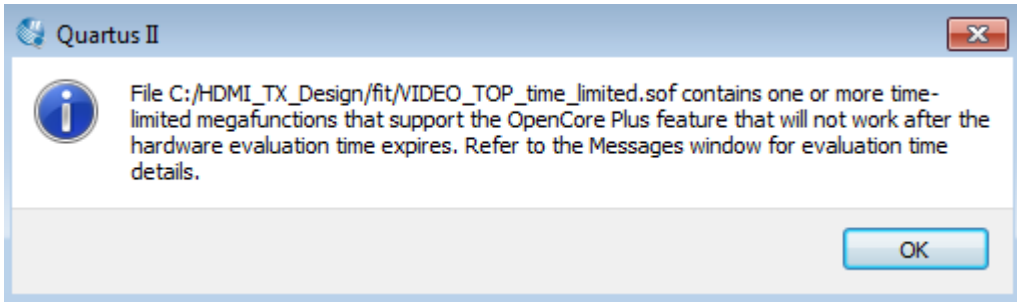
Preparing the Board
(if necessary)

During the compilation of the design, you can prepare the AVDB Board and set up.

- Connect the DisplayPort → HDMI adapter to AVDB DP connector J14.
- Connect an HDMI cable to the above adapter and to a Full-HD screen or TV.
  It must support a resolution of **1920x1080 – 50Hz**
- Connect a mini-USB cable to connector **J10** "*BII*"
- Connect the 12 V DC Power Supply to the connector **J9**.

✔ Once the compilation is performed launch the Quartus II Programmer : 

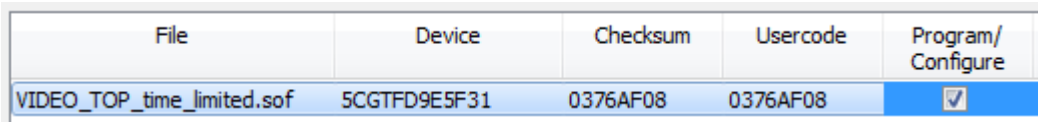
✔ The following message window should appear :



✔ Click on **OK**.

✔ Verify that **Program/Configure** is checked (active) and **Hardware Setup** is correctly configured.
AVDB includes an on-board USB-Blaster II programming adapter !
Click on *Hardware Setup...* if the AVDB programming adapter is not selected already.

NOTE : You may use **remote JTag Programming** if the Instructor has set up a server.



File	Device	Checksum	Usercode	Program/ Configure
VIDEO_TOP_time_limited.sof	5CGTFD9E5F31	0376AF08	0376AF08	☑

✔ Launch the programming by clicking on the icon : 

Once programmed :

- The 4 user Leds (D3, D5, D6, D7) should blink slowly and progressively (heart beat).
- A test pattern should be displayed on the TV screen with a color stripes shifting every second
- This indicates that he design is working correctly.
- The audio stream injected at the line input should be audible at the Headphones connected to the board and also from the TV Speakers.

Design # 2 : HDMI Bypass

This design turns AVDB into a “TV player” that sends Video and Audio to an HDMI TV or monitor.

The video (& audio) come from an HDMI non-encrypted video source player connected to AVDB's HDMI input port.

The TV or monitor HDMI is attached to AVDB's DisplayPort connector using an HDMI adapter.

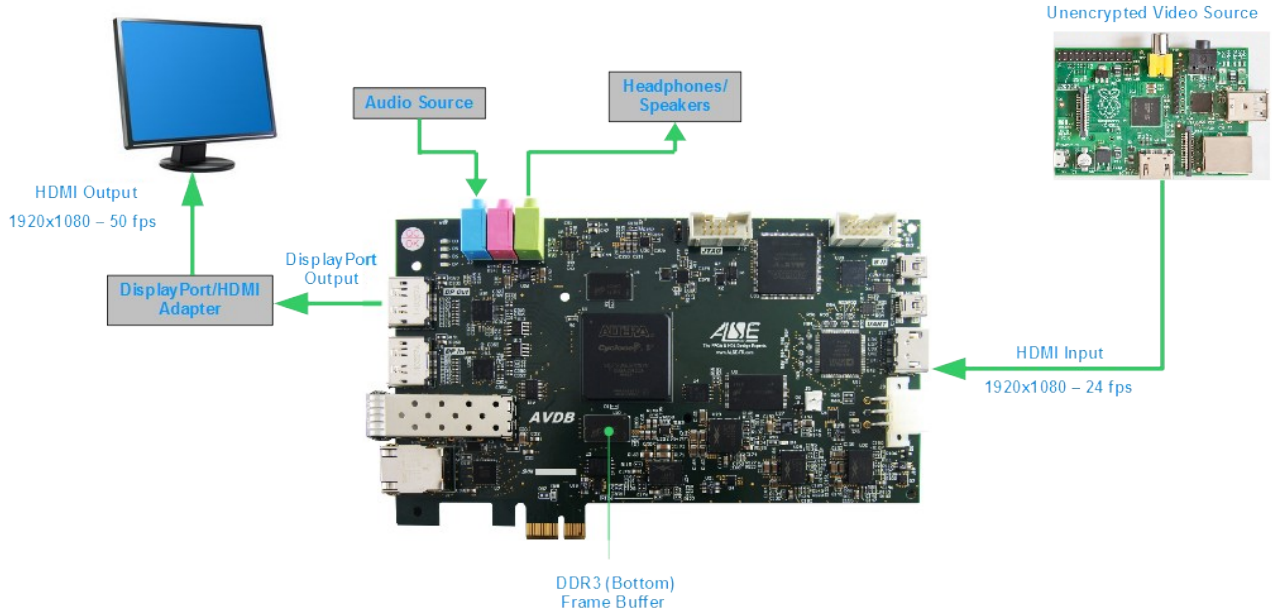
The Audio Codec located on AVDB does play the HDMI source input audio, captures an audio source (if present) and sends it to the TV or Monitor.

The whole design fits in approximately 5,500 ALMs and is built using Qsys.

Design features :

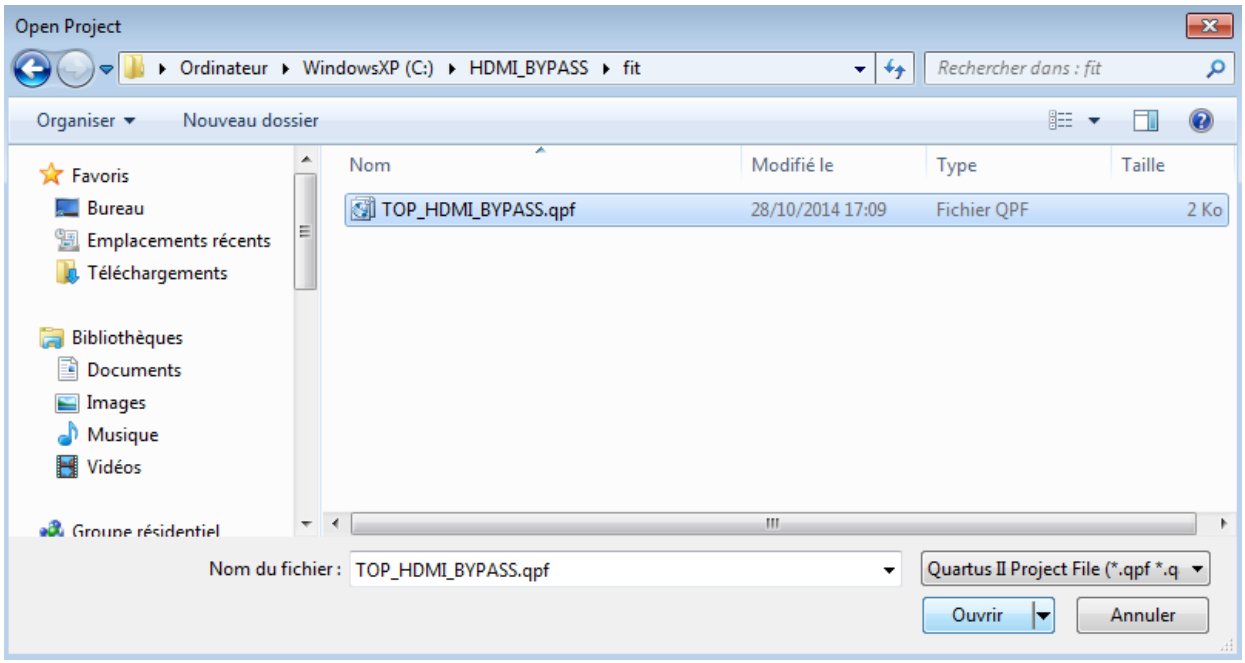
- Video input : non encrypted HDMI input – 1920x1080 @ 24 Fps – RGB on 8 bits
  (we typically use a Raspberry Pi)
- Video Output : HDMI through an adapter – 1920x1080 @ 50 Fps – RGB on 8 bits
- Audio Output : Line Output from the Codec – I2S Mode with Sampling rate @ 48 KHz
- Audio Input : Line input from the Codec – I2S Mode with Sampling rate @ 48 Khz

The Figure below shows the block diagram of the design.



Step-by-Step Instructions

✔ Launch Quartus II and open the project named “**TOP_HDMI_BYPASS.qpf**” located in the HDMI_BYPASS directory.



✔ Proceed as in Design # 1 regarding the OpenCore+ license.

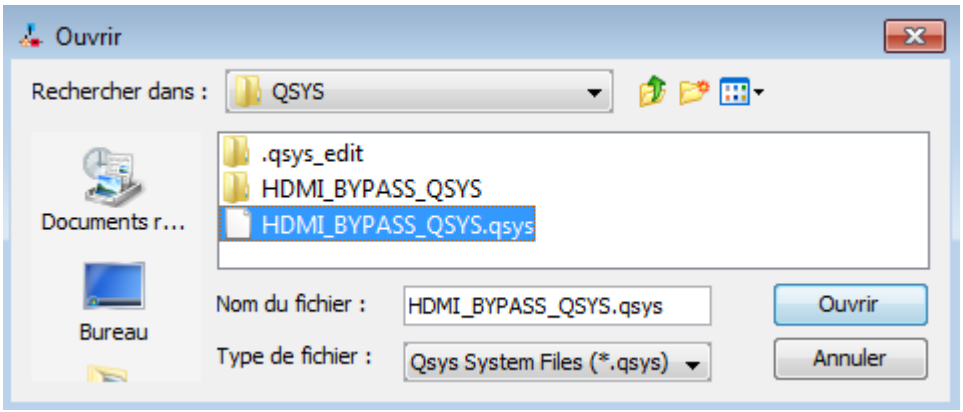
✔ When the new license file location has been added to the current license file a new Licensed function should be visible in the list :

Vendor	Product	Version
ALSE (7D3A)	FFFF	2015.01

✔ Click OK to close this window.

✔ Under Quartus II launch Qsys by clicking on the following icon of the menu (or from the menu Tools → Qsys) :

✔ In Qsys open the component named “VIDEO_BYPASS_QSYS.qsys” located in “../src/qsys/”.



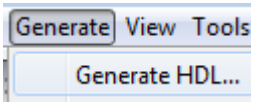
At the question : “Save changes to unnamed ?” → select “Don't save”.

Once opened the system should look like this:

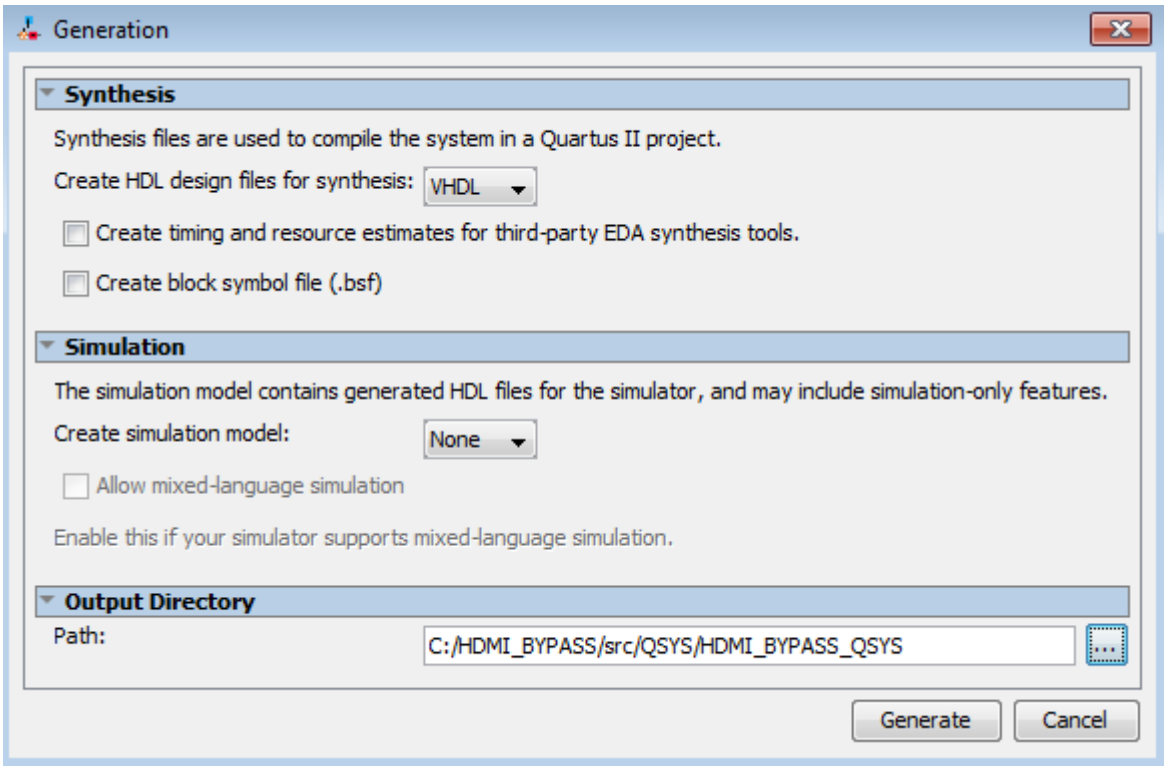
Use	C...	Name	Description	Export	Clock
☑		⊞ i_CLK150	Clock Source		*exported*
☑		⊞ i_MEM_CLK	Clock Source		*exported*
☑		⊞ i_VIDEO_CLK	Clock Source		*exported*
☑		⊞ i_AVDB_CONFIG	AVDB Configuration Block		i_CLK150
☑		⊞ i_HDMI_VIDEO_RX_ST	ALSE HDMI Video Receiver		i_CLK150
☑		⊞ i_FRAME_GEN_ST	ALSE Video Pattern Generator		*multiple*
☑		⊞ i_VIDEO_IN_SEL	ALSE Video Switch		i_VIDEO_CLK
☑		⊞ i_ddr3b_ctrl_hmc	DDR3 SDRAM Controller with UniPHY		*multiple*
☑		⊞ i_FRAME_BUFFER_ST	ALSE FRAME Buffer		*multiple*
☑		⊞ i_HDMI_AUDIO_RX_ST	ALSE HDMI Audio I2S Receiver		i_CLK150
☑		⊞ i_AUDIO_IN_ST	ALSE Audio Generation Block		i_CLK150
☑		⊞ i_AUDIO_IN_SEL	ALSE Audio Switch		i_CLK150
☑		⊞ i_HDMI_TX	ALSE HDMI TX IP		*multiple*

When opening the system, there will be 4 warnings related to the DDR3 Controller.

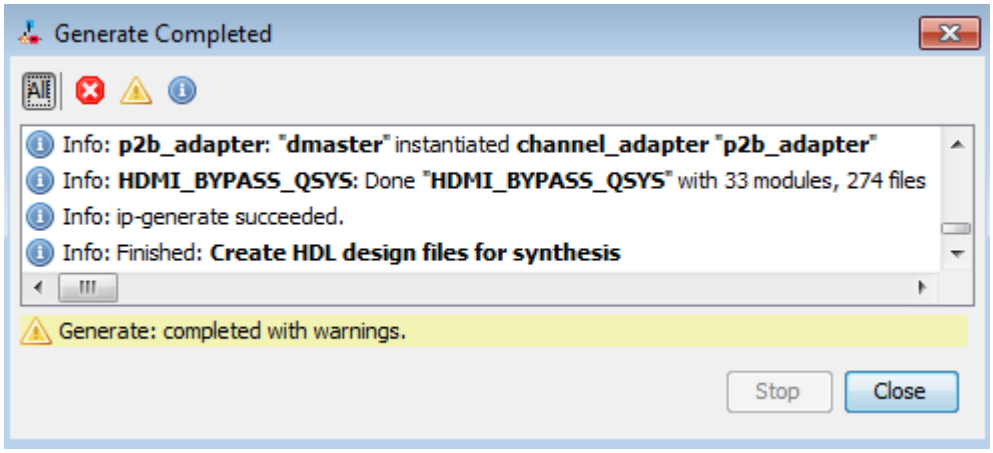
✔ From the menu Generate, choose Generate HDL …



✔ In the Generation Widow, configure options as shown below :



✔ Once generated the following message should appear :



✔ Close Qsys. If a Pop-up window under Quartus II is displayed about the fact that *.qip and *.sip Files have been generated click OK.

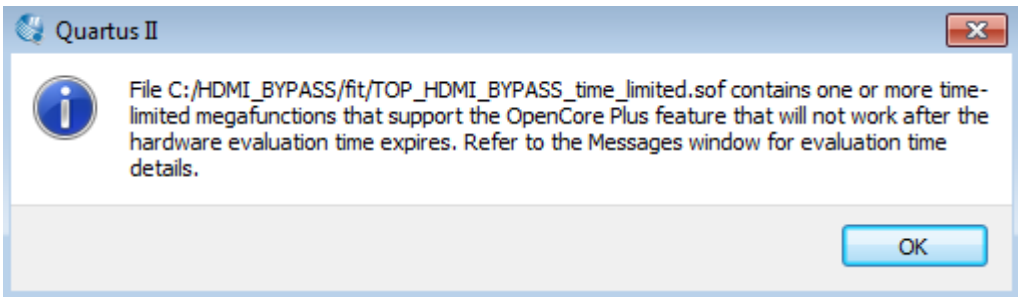
✔ Launch the compilation of the design by clicking on the icon .

✔ During the compilation of the design, prepare the Board.

- Connect a HDMI cable between the Video source and the HDMI Input connector of the Board (J13). Do not power supply the HDMI source yet, it should be done after the FPGA programming.
- Connect a DisplayPort cable with a HDMI adapter from the connector (J14) of the board to a TV screen which can support a resolution of 1920x1080 – 50Hz
- Connect an audio source to the audio line input connector (J2) of the board with an audio jack cable
- Optionally Connect headphones or speakers with to the audio line output connector (J4) of the board with an audio jack
- connect a USB cable to the connector J11(JTAG) or J12(Blaster II with a mini USB Cable)
- Connect the 12 V DC Power Supply to the connector J9.

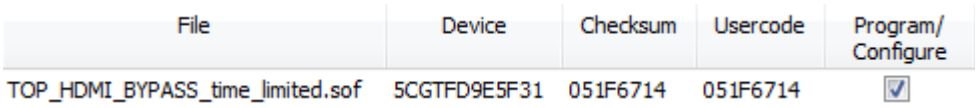
✔ Once the compilation is performed click on the icon :

✔ The following window should appear :



✔ Click on OK.

✔ Verify that Program/Configure is checked and that the Hardware Setup is correctly configured. If it is not the case click on Hardware Setup... to select the connection to use to program the board.



File	Device	Checksum	Usercode	Program/ Configure
TOP_HDMI_BYPASS_time_limited.sof	5CGTFD9E5F31	051F6714	051F6714	☑

✔ Launch the programming by clicking on the icon :

Once programmed :

- The 4 user LEDs (D3, D5, D6, D7) should blink slowly and progressively (“heart beat”).
- Video : a test picture with ALSE's logo should be displayed during a few seconds. When you see the ALSE's logo power supply the HDMI source. The Display screen should show the video read from the Video source in the appropriated format.
- Audio : the audio input from the Video source should be available through the Display Screen speakers and also on the Board Headphone Speakers. If an audio source is connected to the Codec **Line In**put connector of the Board, it will replace the HDMI audio stream and be sent to the Video Display (and to the headphones / speakers).

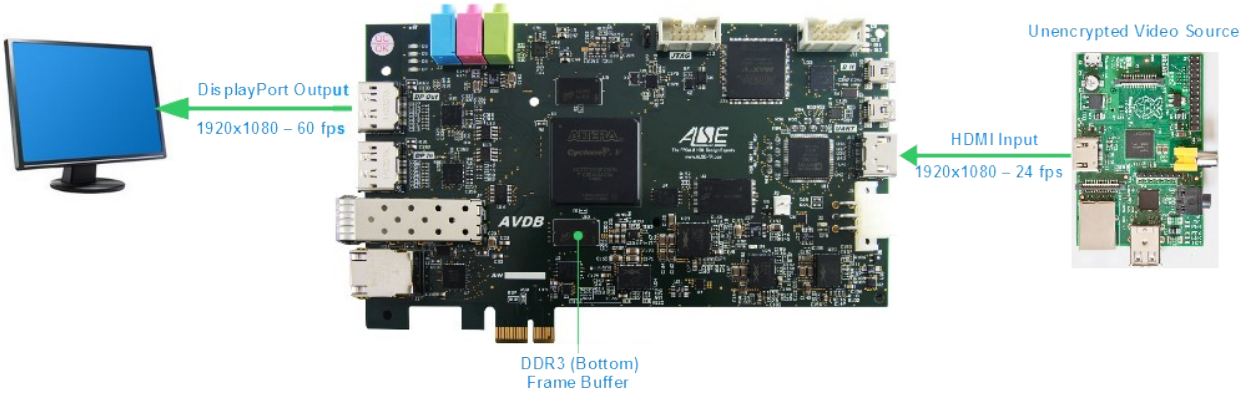
Design # 3 : HDMI to DisplayPort

This design is a variant of the HDMI Bypass demo, with the Video output stream being displayed using the **DisplayPort** Video Standard.

Design features :

- Video input : non encrypted HDMI input – 1920x1080 @ 24 Fps – RGB on 8 bits
  (we typically use a raspberry Pi)
- Video Output : DisplayPort – 1920x1080 @ 60 Fps – RGB on 8 bits

The Figure below shows the block diagram of the design.



We will not provide all detailed instructions (they are similar to the previous Reference designs).

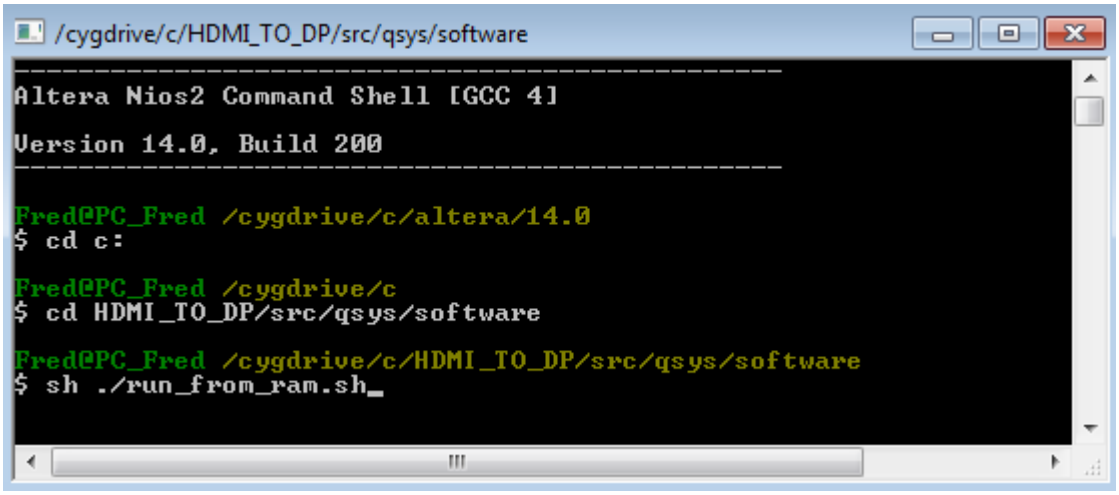
Qsys system

Use	C...	Name	Description	Export	Clock	Base	End	IRQ
☑		**merged_resets**	Reset Bridge					
☑		**clk_vip**	Clock Source		***exported***			
☑		**clk_aux**	Clock Source		***exported***			
☑		**clk**	Clock Source		***exported***			
☑		**onchip_mem**	On-Chip Memory (RAM or ROM)		**clk**	0x0002_0000	0x0002_ffff	
☑		**cpu**	Nios II Processor		**clk**	0x0004_2000	0x0004_27ff	
☑		**jtag_uart**	JTAG UART		**clk**	0x0004_2a80	0x0004_2a87	16
☑		**sys_clock_timer**	Interval Timer		**clk**	0x0004_2800	0x0004_281f	1
☑		**sysid**	System ID Peripheral		**clk**	0x0004_2b30	0x0004_2b37	
☑		**msa_dump_pio**	PIO (Parallel I/O)		**clk**	0x0004_2980	0x0004_298f	
☑		**xdash**	PIO (Parallel I/O)		**clk**	0x0004_2900	0x0004_290f	
☑		**i2c**	oc_i2c_master		**clk**	0x0004_2ac0	0x0004_2adf	3
☑		**dp**	DisplayPort		***multiple***	0x0004_1000	0x0004_17ff	0
☑		**aux_tx_debug_fifo**	Avalon FIFO Memory		**clk_aux**	multiple	multiple	
☑		**cvo2**	Clocked Video Output II (4K Ready)		**clk_vip**	0x0008_1000	0x0008_13ff	4
☑		**i_FRAME_GEN_ST**	ALSE Video Pattern Generator		***multiple***	0x0008_0000	0x0008_001f	
☑		**i_HDMI_VIDEO_RX_ST**	ALSE HDMI Video Receiver		**clk_vip**			
☑		**i_FRAME_BUFFER_ST**	ALSE FRAME Buffer		***multiple***			
☑		**i_ddr3b_ctrl_hmc**	DDR3 SDRAM Controller with UniPHY		***multiple***	multiple	multiple	
☑		**i_AVDB_CONFIG**	AVDB Configuration Block		**clk**			
☑		**i_VIDEO_IN_SEL**	ALSE Video Switch		**clk_vip**			

Nios II system

In this design, a Nios II embedded processor is used to dynamically ocnfigure some IP blocks.

✔ Open a NIOS II Command Shell terminal and go to the software directory located in ./src/qsys :



✔ Execute the script “run_from_ram.sh” in order to boot the NIOS II from the internal memory of the FPGA.

✔ Launch the compilation of the design.

Prepare the board

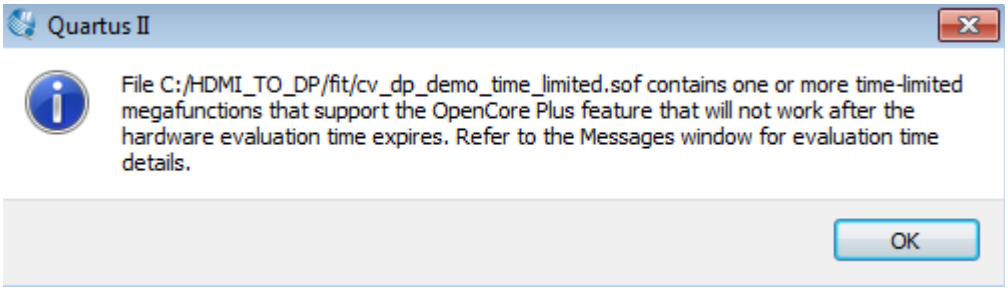
✔ During the compilation of the design, prepare the Board :

- Connect a HDMI cable between the Video source and the HDMI Input connector of the Board (J13). Do not power supply the HDMI source yet, it should be done after the FPGA programming.
- Connect a DisplayPort cable from J14 connector to a DisplayPort screen which can support a resolution of 1920x1080 – 60Hz
- Optionally, connect headphones or speakers to J4 audio line output connector.
- Connect a mini-USB cable to the Jtag Blaster II connector J12 (*BII*)
- Connect the 12 V DC Power Supply to the connector J9.

Test !

✔ Once the compilation is performed click on the icon :

✔ The following window should appear :



Click on OK.

✔ Verify that Program/Configure is checked and that the Hardware Setup is correctly configured. If it is not the case click on Hardware Setup... to select the connection to use to program the board.

File	Device	Checksum	Usercode	Program/ Configure
cv_dp_demo_time_limited.sof	5CGTFD9E5F31	06B65720	06B65720	☑

✔ Launch the programming by clicking on the icon : 

Once programmed :

- The 4 user LEDs (D3, D5, D6, D7) should blink slowly and progressively (“heart beat”).
- Video : a test picture with ALSE's logo should be displayed during a few seconds.
  When you see the ALSE's logo, apply power to the HDMI source (Raspberry Pi).
  The Display screen should show the video read from the Video source in the appropriate format.
- Audio : the audio input from the Video source should be output to the Headphones / Speakers.

Design # 4

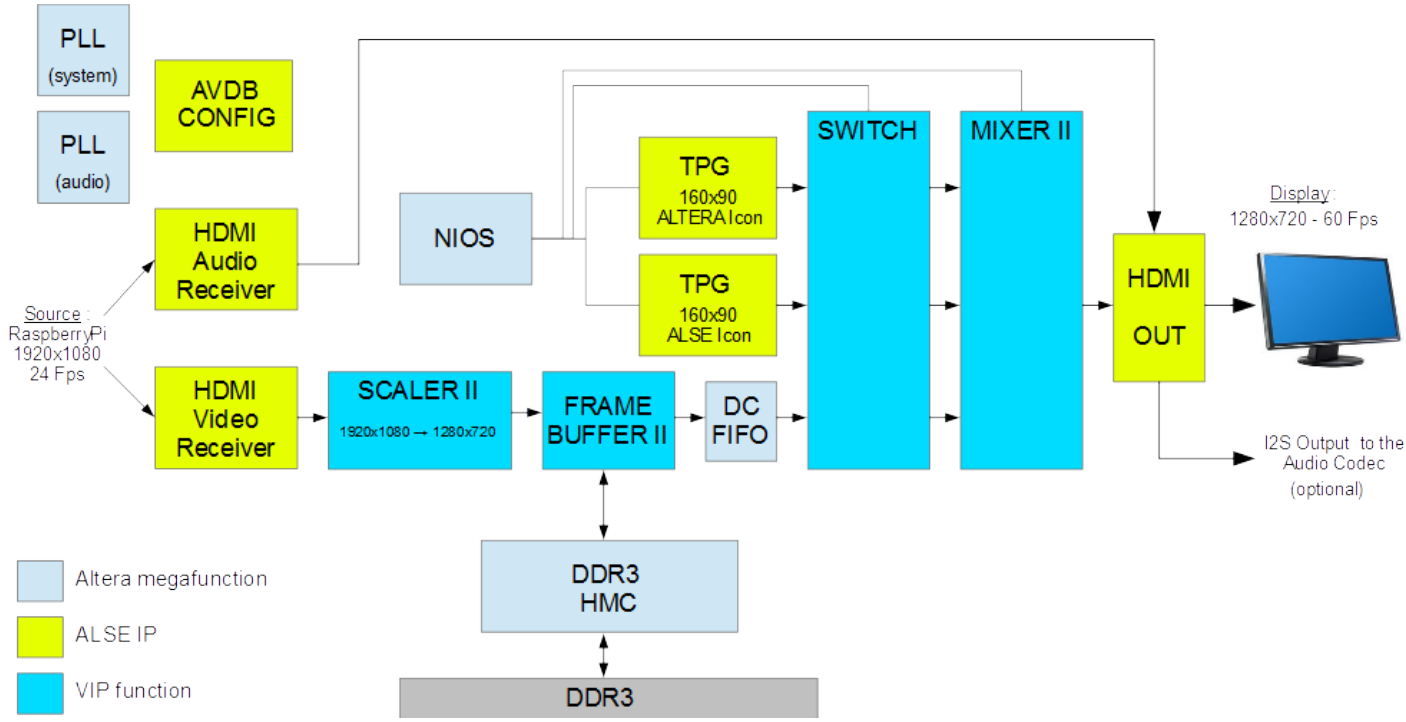
AVDB VIP Reference Design
aka “VIP-based HDMI Design Demo”

The FPGA & HDL Design Experts
www.ALSE-FR.com

Introduction

This design is not spectacular to look at, but it does actually implement a complete and functional “Video Pipeline” using several IPs. A separate and detailed set of instructions is available ('Lab4”) to demonstrates how to build such a system, from scratch, and step-by-step.

Design Description



As seen in the above block diagram, this design uses several functions of the Video and Image Processing (VIP) suite proposed by Altera, as well as a number of ALSE IPs.

IPs from the VIP suite :

- Scaler II
- Frame Buffer II
- Switch
- Mixer II

ALSE IPs :

- AVDB Configuration block
- Test Pattern Generator customized to generate Altera & ALSE logos.
- HDMI Receiver (Video & Audio)
- HDMI Output

Design Inputs & Outputs

The FPGA should be loaded with this design, a Display screen or TV with HDMI input and supporting at least the 1280 x 720p resolution should be connected to the DP to HDMI conversion cable, and a Full-HD non-encrypted video source should be connected to the HDMI-in connector.
The screen should display the video input, downsized and with a higher rate, and with two logos superimposed at varying positions.

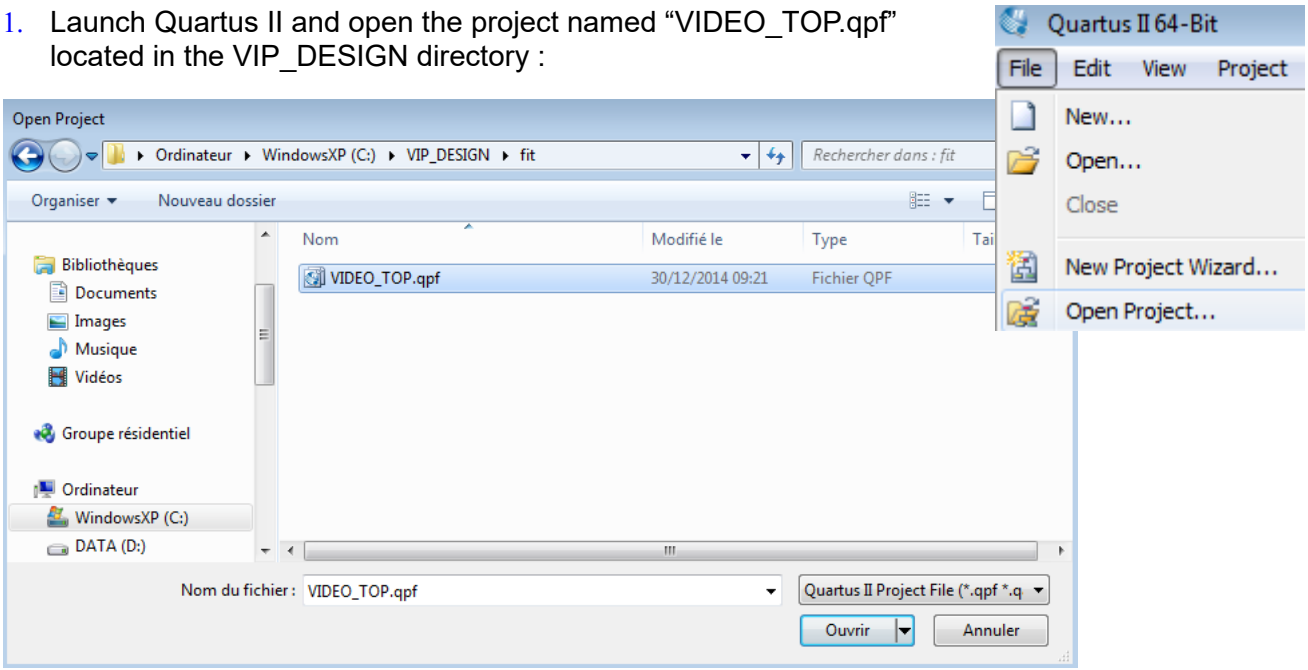
External Video Input	HDMI Full-HD – 1920x1080p – RGB – 24 Fps, from the Raspberry Pi (eg)
Video Output	HDMI – 1280x720p – RGB – 60 Fps
Additional Audio Output	I2S Output to the Codec *Line Out* connector

The NIOS processor embedded in the design is used to display and move every second the Altera & ALSE logos all around the output screen.

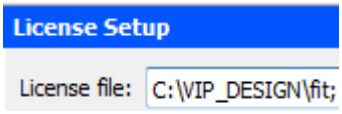
Preparing the project

You need to activate ALSE's OpenCorePlus license to enable the compilation of this project.

1. Launch Quartus II and open the project named “VIDEO_TOP.qpf” located in the VIP_DESIGN directory :



2. Once the is opened, activate the ALSE license file.
Menu : **Tools** > **License setup …**
3. In the *License File* area, **insert the directory** of the project **at the beginning, followed by a semicolon**. The license file “alse_opencore_xxx.ocp” is located in this directory.
In the case below, we unpacked the project under C:\VIP_DESIGN :



4. When the correct directory has been added to the current license file information, a new Licensed function should be visible in the list →

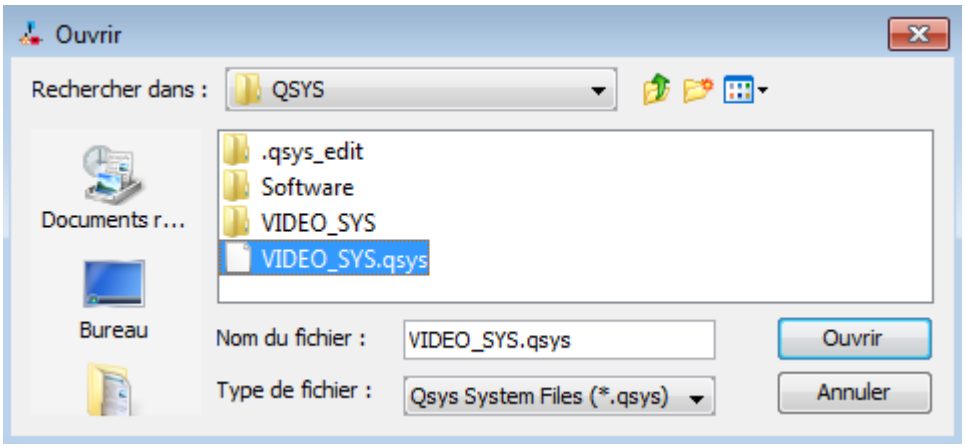
Vendor	Product	Version
ALSE (7D3A)	FFFF	2015.01

5. Click OK to close this licensing tools window.

Compiling the Project

6. Under Quartus II launch **Qsys** by clicking on the following icon of the menu (or from the menu Tools → Qsys) :

7. In Qsys, open the component named “VIDEO_SYS.qsys” located in “../src/QSYS/”.



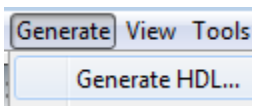
8. At the question : “*Save changes to unnamed ?*” → select “**Don't save**”.
9. Once opened, the Qsys system should look like :

Use	C...	Name	Description	Export	Clock	Base	End	IRQ
☑		SYS_CLK	Clock Source		*exported*			
☑		BOARD_CONFIG	AVDB Configuration Block		SYS_CLK			
☑		HDMI_VIDEO_IN	ALSE HDMI Video Receiver		SYS_CLK			
☑		HDMI_AUDIO_IN	ALSE HDMI Audio I2S Receiver		SYS_CLK			
☑		VIDEO_IN_TO_SCALER	Avalon-ST Timing Adapter		SYS_CLK			
☑		SCALER	Scaler II - Edge Adaptive		SYS_CLK			
☑		VIDEO_BUFFER	Frame Buffer II (4K Ready)		SYS_CLK			
☑		DDR3_HMC	DDR3 SDRAM Controller with UniPHY		*multiple*	multiple	multiple	
☑		BUFFER_TO_FIFO	Avalon-ST Timing Adapter		SYS_CLK			
☑		DVI_FIFO	Avalon-ST Dual Clock FIFO		*multiple*			
☑		VID_CLK	Clock Source		*exported*			
☑		FIFO_TO_SWITCH	Avalon-ST Timing Adapter		VID_CLK			
☑		ALTERA_ICON	ALSE Icon Generator		VID_CLK			
☑		ALSE_ICON	ALSE Icon Generator		VID_CLK			
☑		SWITCH	Switch		VID_CLK	0x0000_9100	0x0000_917f	
☑		MIXER	Mixer II (4K Ready)		VID_CLK	0x0000_9000	0x0000_90ff	
☑		MIXER_TO_HDMI	Avalon-ST Timing Adapter		VID_CLK			
☑		HDMI_OUT	ALSE HDMI TX IP		*multiple*			
☑		SYS_ID	System ID Peripheral		VID_CLK	0x0000_9180	0x0000_9187	
☑		JTAG_UART	JTAG UART		VID_CLK	0x0000_9188	0x0000_918f	0
☑		CPU_MEM	On-Chip Memory (RAM or ROM)		VID_CLK	0x0000_4000	0x0000_7fff	
☑		CPU	Nios II Gen2 Processor		VID_CLK	0x0000_0800	0x0000_0fff	

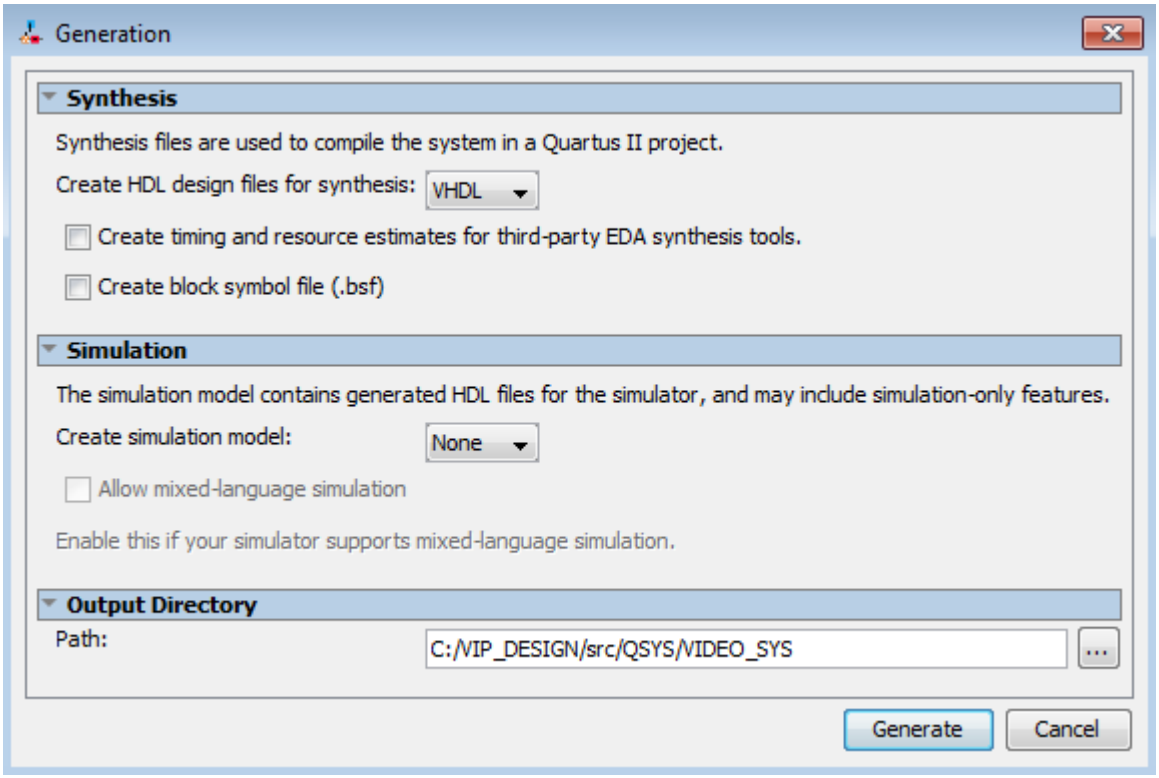
Note: you may get 5 warnings (4 related to the DDR3 Controller and 1 related to the Switch).

If you are familiar with Qsys, you can explore the system, but do not change anything in it !

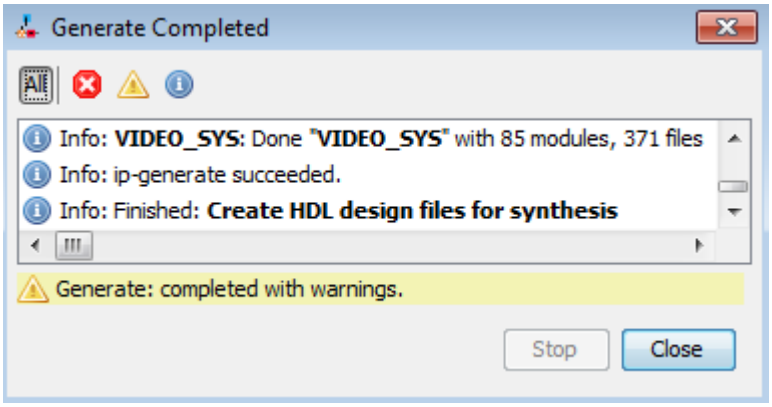
10. From the menu **Generate**, choose **Generate HDL …**



11. In the *Generation* window configure options as shown below :



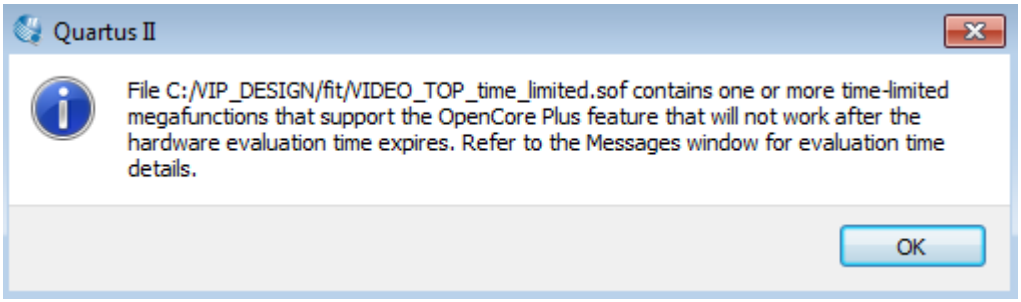
12. Once generated the following message should appear :



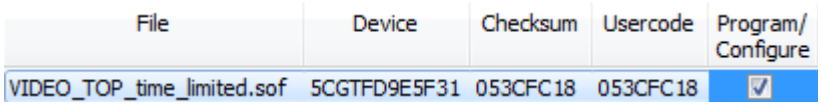
13. Close Qsys. If a Pop-up window under Quartus II is displayed about the fact that *.qip and *.sip files have been generated, click on OK.
14. Launch the compilation of the design by clicking on the “Start Compilation” icon or use the Menu. (The compilation for this design takes about 15 minutes, depending on the computer).

Running on AVDB

15. During the compilation of the design, prepare the AVDB board, if not done already.
    - Connect a HDMI cable between the Video source and the HDMI Input connector of the Board (J13). Do not power the HDMI source yet (or if powered, do not connect it yet), it's better to wait until after the FPGA is programmed with the design.
    - Connect a DisplayPort to HDMI *adapter cable* to connector (J14). Connect it with an HDMI cable to a TV or a monitor that can support 1280x720 – 60Hz resolution (HD-ready is enough).
    - Connect a mini-USB cable to connector J12 “BII” (Blaster II).
    - Connect the AVDB 12 V DC Power Supply to connector J9. Power up AVDB.
16. Once the compilation is performed, click on the  icon to program the FPGA.
17. The following window should appear :



18. Click on OK.
19. Verify that Program/Configure is checked and that the Hardware Setup is correctly set. If not,click on *Hardware Setup...* to select the AVDB on-board Blaster II.



20. Launch the programming by clicking on the Start icon :

21. Once programmed :
    - The 4 User LEDs (D3, D5, D6, D7) should blink slowly (“heart beat”).
    - Video : a test picture with ALSE's logo should be displayed during a few seconds.
      When you see ALSE's logo and Altera's logo moving on the screen, you can power on or connect the HDMI source. The display screen should show the Video source in the downsized format with the two logos moving.
    - Audio : the TV (or monitor) should play the audio stream generated by the HDMI source.
      The audio is also available (if enabled) from the Line Out connector (through the Codec).

Congratulation !
You have successfully compiled
a complete Video Project
and tested it on AVDB.

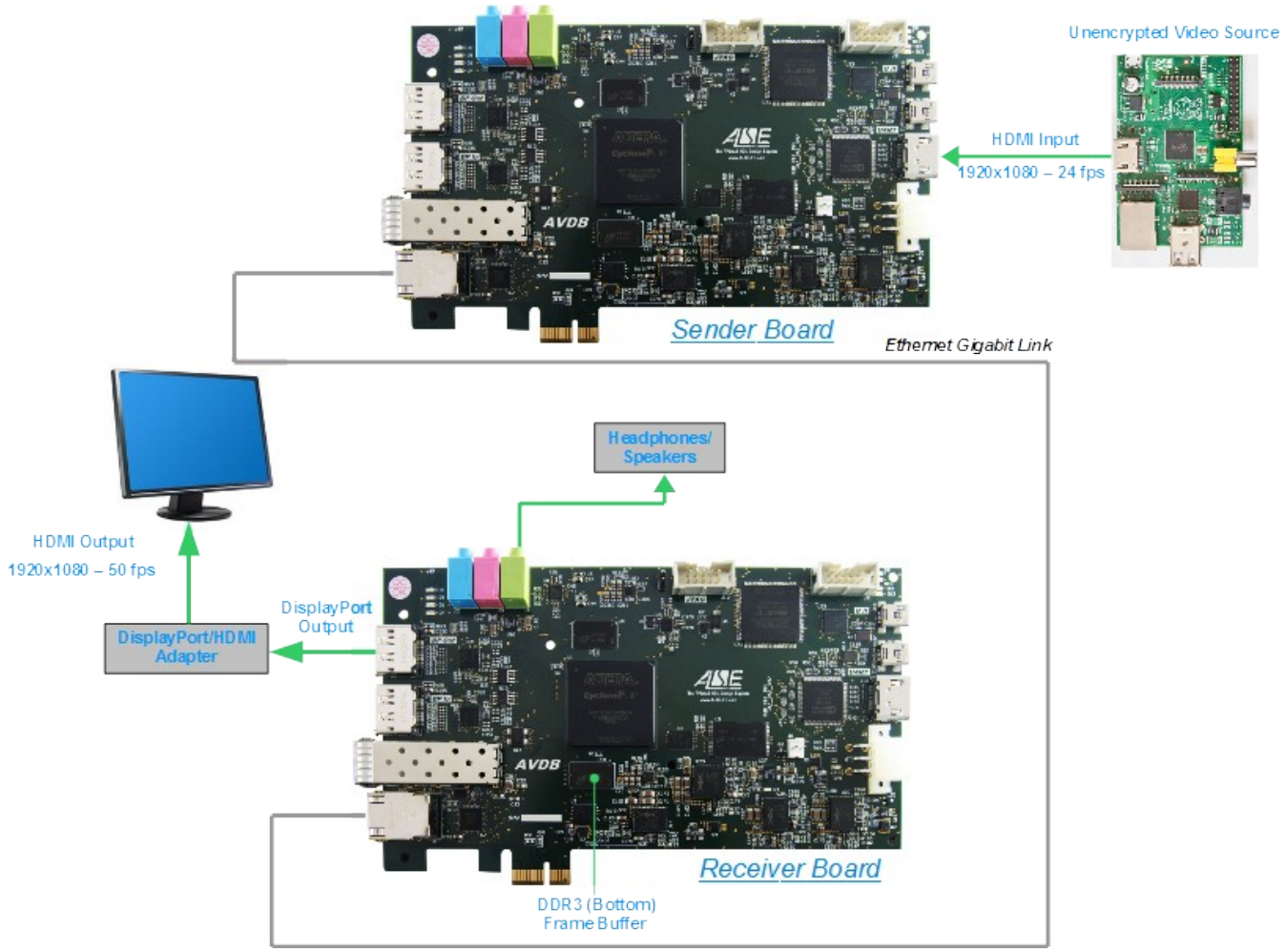
Design # 5 :Video Streaming over Ethernet (4:2:0, no compression)

In this design, a Full HD video stream (reduced to 4:2:0) is transmitted over Gigabit Ethernet using the **GEDEK** IP from ALSE (*Gigabit Data Exchange Kit*).

This demo illustrates the efficiency and throughput of GEDEK: the streamed data represents close to 90% of the theoretical Gigabit bandwidth.

Two AVDB Kits are required for this demo :

- AVDB 1 (Sender) : receives a Full HD Video Stream on the HDMI Input Interface and transmits the video contents over Ethernet to the second AVDB.
- AVDB 2 (Receiver) : receives the Video Stream over Ethernet and displays it on the DisplayPort/HDMI Interface through a Frame Buffer (allowing Frame rate control).



NB : Please note that this setup also exists :

- With a JPEG encoder at input and a JPEG decoder in the receiver in order to reduce the Ethernet bandwidth used over the network.
- With the NEW **Wavelet Encoder/Decoder IPs developed by ALSE.** This new Codec provides extremely high quality and a very good compression rate.

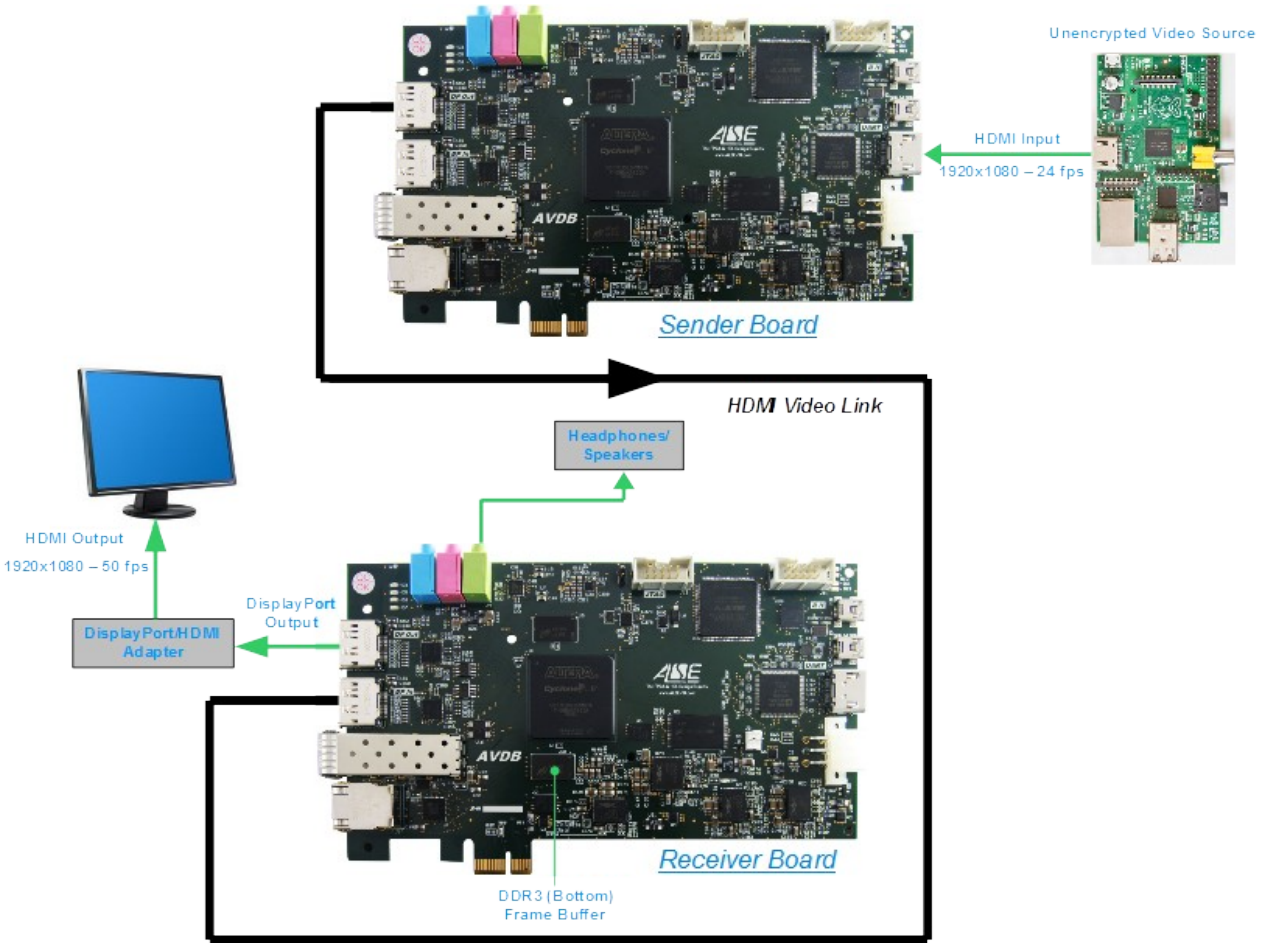
Please Contact ALSE if you are interested by this Design and the demo files.

Design # 6 : Video Streaming over HDMI

This demonstration is a variant of the Video Streaming demo over Ethernet except that in this case the video stream is transmitted from one board to the other through the DisplayPort Physical Link and connectors, but the logical format and encoding is **HDMI**.

This setup has been developed on AVDB to test and demonstrate ALSE's **transceiver-based HDMI receiver IP.**

Just like the Ethernet streaming demo video, the audio is also transported from the source to the TV or monitor through all IPs.



Please Contact ALSE if you are interested by this Design and the demo files.

Design # 7 : Video Streaming over Ethernet with Wavelet Compression - Decompression

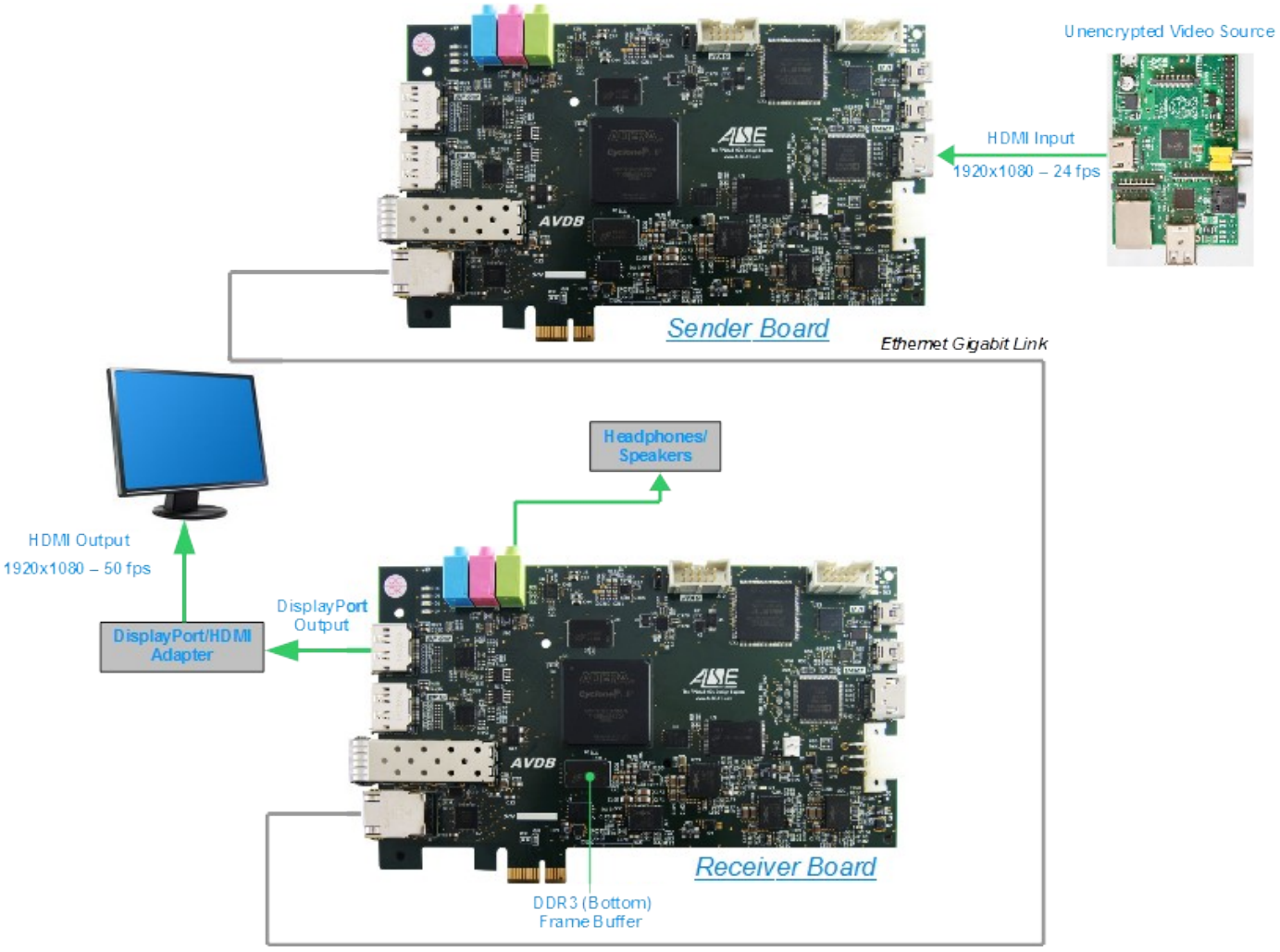
In this design, a Full HD video stream (HDMI input) is received and compressed using ALSE's new **Wavelet Coder**. The compression rate is high (close to 30) so the transmission over Gigabit Ethernet through the **GEDEK** IP occupies less than 8% of the theoretical GB bandwidth, thus allowing the use of Fast (100M Ethernet), or allowing to stream more than 10 x HD videos simultaneously !

With its adaptive compression, it can convert a 1920x1080 30fps raw video stream (about 1.5 Gbps) into a ~50 Mbps compressed stream. The whole Codec can also be fully parametrized to reach optimal compression and quality, with more than 64 parameters.

A software version of the Codec also enables you to stream compressed file from a computer to a decoder, or to record a compressed file streamed by the encoder.

Two AVDB Kits are required for this demo:

- AVDB 1 (Sender): receives a Full HD Video Stream on the HDMI Input Interface and transmits the video contents over Ethernet to the second AVDB.
- AVDB 2 (Receiver): receives the Video Stream over Ethernet and displays it on the DisplayPort/HDMI Interface through a Frame Buffer (allowing Frame rate control).



Please Contact A.L.S.E if you are interested by this Design or by ALSE's Wavelet Codec.

Design # 8 : Altera “Best In Class” PCIexpress

We have a working Reference Design demonstrating our port of Altera's “Best In Class PCIexpress IP” to Cyclone V GT.

This design is a PCIe gen2 x1 Qsys system demonstrating bidirectional DMA transfers between FPGA and HOST Memory. Driver and demo software is provided for Linux 2.6 kernel and windows 7 x64.

It is configured using a dedicated BAR issuing register configuration through a convenient host API.

Please contact ALSE for more details.

Design # 9 : Ultra-High Definition (4K) Video Demo

When we designed 2AVDB, we intentionally decided to not position target Ultra-High Definition Video also known as “4K”. *However*, we found that it was possible to implement complete applications working at these extreme resolutions (but within compatibility with HDMI 1.4).

This design example demonstrates 4K resolutions both in the Video input and the video output.

Please contact ALSE for more details.

ALSE Intellectual Property Blocks

In our Reference Designs, we use the following ALSE IPs :

- ALSE HDMI input through ADV7619
- ALSE HDMI in (with I2S audio out)
- ALSE HDMI out (with I2S input)
- GEDEK (Gigabit Ethernet hardware communication stack used for Video streaming)
- I2S Codec (SSM2603) Interface
- I2C controller (multiple instances)
- NAND Flash Memory Controller with BCH ECC
- Quad-SPI Controller
- JPEG video compression engine
- JPEG video decompression engine
- Wavelet video compression engine
- Wavelet video decompression engine
- DDR3 Frame Buffer (with rate changing)
- Video Test Pattern Generator
- Upscale / Downscale resizers
- etc

-=oOo=-