DE1_SOC music player demo

Many commercial media/audio players use a large external storage device, such as an SD Card or CF card, to store music or video files. Such players may also include high-quality DAC devices so that good audio quality can be produced. DE1_SoC board provides the hardware and software needed for Micro SD Card access and professional audio performance so that it is possible to design advanced multimedia products using the DE1_SOC board. It is very convenient to get music file is the micro SD card with the embedded linux operate system and file system. This demonstration presents using HPS to control the FPGA to play music on De1_SoC board.

Function Block Diagram

As it known to the users that the micro SD card is accessible to the HPS and the audio chip is accessible to the FPGA part on De1_SOC board. So the HPS must translate date to the FPGA to control the audio chip and play music in this demonstration. In the diagram of this demonstration shown in Figure 6-3, The Lightweight HPS-to-FPGA Bridge works as the media to translate date from HPS to FPGA.

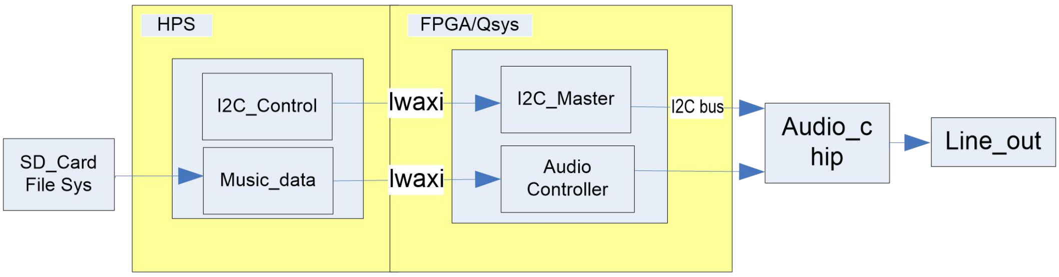


Figure 6-3 block diagram of music player demonstration

The data translated on Lightweight HPS-to-FPGA Bridge is converted into Avalon_MM master interface in FPGA part. The I2C_master IP and Audio Controller worked as Avalon slave component. The I2C_master IP control the registers in audio decode chip through I2C bus and audio Controller translate date to the audio decode chip.

Music player control

The program running in hps access the Lightweight HPS-to-FPGA Bridge and translate date through it to the fpga. Since the user application program can't control the peripheral through the physical address, so the lwaxi bridge should be mapped into the user space. The method to get music data is so simply that you just need to use the file operate functions such as fopen( ) and fread( ) in the standard C library.

The audio decode chip WM8731 has a I2C control interface to write the register in it.

The audio chip should be configured before sending audio signal to the audio chip. The program uses I2C protocol to configure the audio chip working in master mode; the audio output interface working in I2S 16-bits per channel and with sampling rate according to the wave file contents. In audio playing loop, the main program reads 512-byte audio data from the micro SD Card, and then writes the data to DAC FIFO in the Audio Controller. Before writing the data to the FIFO through lwaxi bridge, the program will verify if the FIFO is full. The design also mixes the audio signal from the microphone-in and line-in for the Karaoke-style effects by enabling the BYPASS and SITETONE functions in the audio chip.

■ Demonstration Source Code

- Build tool: Altera SoC EDS V13.0
- Project directory: \Demonstration\ de1_soc_audio_music.
- Quick file directory:\ Demonstration\ de1_soc_audio_music \ quickfile
- Batch File :Demonstration\de1_soc_audio_musicquickfile\sof_dwonload\test.bat
- FPGA Configure File : DE1_SOC_golden_top.sof
- Binary file: de1_soc_audio_app
- Build app command: make ('make clean' to remove all temporal files)
- Execute app command: ./de1_soc_audio_app –f My_Heart_Will_Go_On.wav (or ./de1_soc_audio_app –f My_Heart_Will_Go_On.mp3)

Demonstration Setup

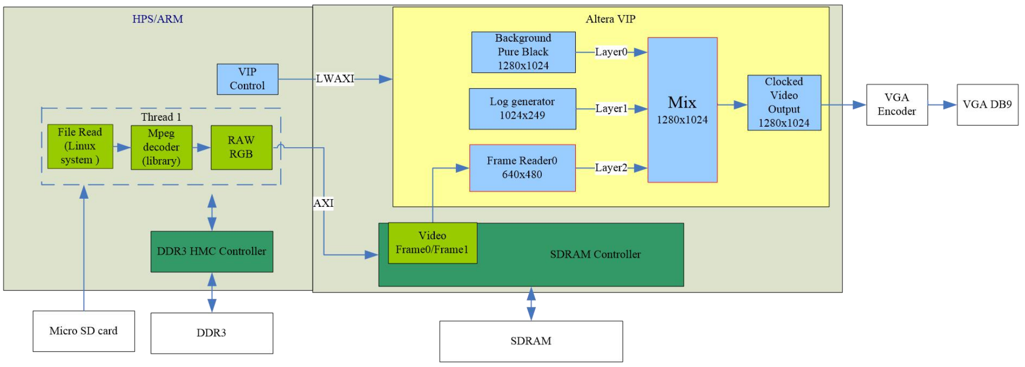
- Make sure Quartus II and Nios II are installed on your PC.
- Connect the USB blaster cable to the USB blaster connector (J13) on the DE1_SOC board and host PC install USB Blaster driver II if necessary.
- Connect the USB cable to the USB-to-UART connector (J4) on the DE1_SOC board and host PC.
- Make sure the executable file " de1_soc_audio_app " and the .mp3 or .wav music file is copied into the SD card under the "**/home/root**" folder in Linux.
- Insert the booting micro SD card into the DE1_SOC board.
- Power on the DE1_SOC board.
- Launch PuTTY to connect to the UART port of DE1_SOC board and type "**root**" to login Altera Yocto Linux.
- Execute the demo batch file “test.bat” under the \ quickfile \sof_dwonload
- In the UART terminal of PuTTY, execute "./de1_soc_audio_app –f My_Heart_Will_Go_On.wav (or .mp3)" to start the program.
- Insert a speaker to the line out connector and you should listen the music. Press "CTRL + C" to terminate the program.

DE1_SOC video decode demonstration

It is very fashion to display video in multimedia products. The fpga and arm processor have special advantage in video processing because their high speed and programmable logic in embedded appication. This demonstration presents how to use HPS to decode mpeg2 format video and display it on the VGA monitor on DE1_SoC board.

Function Block Diagram

Figure 6-4 show the block diagram of the video decode demo. The HPS use both LWAXI bridge and AXI bridge to communicate with FPGA. The LWAXI bridge is used for translating vip suite controlling signal and the axi bridge is used for translating video data .Since the axi bridge data width can be configured as 128 bits at most and the lwaxi bridge is only 64 bits, the axi bridge is more suitable for translating large mount of data. The sdram on fpga part is used storage frame data and the frame reader read the data. The pattern generator ip and log generate ip constitute the background of demonstration through the mix ip. The output clock generate ip get the data from mix and generate the VGA timing data. In this demonstration, 640x480 resolution format video is used and the background is 1280x1024 pixels.



MPEG: File format: *640x480.m2v

Figure 6-4 diagram of video decode demonstration

Video decode control

The program running in hps access the Lightweight HPS-to-FPGA Bridge and HPS-to-FPGA Bridge then translate date to the fpga. Since the user application program can't control the peripheral through the physical address, so the lwaxi bridge and axi bridge should be mapped into the user space. The method to get video data is so simply that you just need to use the file operate functions such as fopen() and fread() in the standard C library.

In this demonstration, we use the mpeg2 format video to display and the hps should decode the video data into raw RGB format data before translating it to the FPGA. We add the libmpeg library to our application to decode the video data. The library supply us the interface function to decode the video. We just need to call the function in you application program and add the library when compile the program.

■ Demonstration Source Code

- Build tool: Altera SoC EDS V13.0
- Project directory: \Demonstration\ vip_demo_vedio
- Quick file directory:\ Demonstration\ vip_demo_vedio \ quickfile
- Batch File: \ Demonstration\ vip_demo_vedio\ quickfile \sof_dwonload\test.bat,
- FPGA Configure File : DE1_SOC_golden_top.sof
- Binary file: de1_soc_vip_mpeg
- Build app command: make ('make clean' to remove all temporal files)
- Execute app command: ./ de1_soc_vip_mpeg v_640x480.m2v

Demonstration Setup

- Make sure Quartus II and Nios II are installed on your PC.
- Connect the USB blaster cable to the USB blaster connector (J13) on the DE1_SOC board and host PC install USB Blaster driver II if necessary.
- Connect the USB cable to the USB-to-UART connector (J4) on the DE1_SOC board and host PC.
- Make sure the executable file " de1_soc_vip_mpeg " and the .m2v video file is copied into the SD card under the "**/home/root**" folder in Linux.
- Insert the booting micro SD card into the DE1_SOC board.
- Power on the DE1_SOC board.
- Launch PuTTY to connect to the UART port of SoCKit board and type "**root**" to login Altera Yocto Linux.
- Execute the demo batch file "test.bat" under the \ quickfile \sof_dwonload
- In the UART terminal of PuTTY, execute ".\de1_soc_vip_mpeg v_640x480.m2v " to start the program.
- Connect a VGA monitor to the VGA connector and you will find a video will display on the monitor ,and the video will move around. Press "CTRL + C" to terminate the program.