Application Note

Akademie věd České republiky
Ústav teorie informace a automatizace AV ČR, v.v.i.

EMC^2-DP HDMI in HDMI out Platform

Lukas Kohout, Zdenek Pohl, Jiri Kadlec
[kohoutl, zdenek.pohl, kadlec]@utia.cas.cz

Revision history

Rev.	Date	Author	Description
0	11.07.2016	L. Kohout	Desciprion of platform
1	08.08.2016	L. Kohout	Removed full HW project
2			

Contents

Acknowledgement

This work has been supported from project EMC^2, project number ARTEMIS JU 621429 and MSMT 7H14005.

1. Introduction

This document provides description of UTIA HDMI in HDMI out video demonstrator based on Sundance EMC^2-DP carrier, Trenz System on Module SoM based on Xilinx Zynq and Avnet FMC extension card. The text describes standalone and Petalinux solutions.

2. Description

The HDMI in HDMI out video platform consists of the following components:

- Hardware built from Sundance EMC^2-DP v2 carrier board [1], FPGA SoM Trenz TE0715-7030 [2] and FMC IMAGEON extension card [3], see Figure 1.

  IMPORTANT NOTE: EMC^2-DP v2 carrier board requires modifications to run the video demo. FMC pins G2 and G3 need to be swapped, it is recommended to do this modification before the SoM is plugged in. For those who plan to modify the carrier on their own, please do not hesitate to contact kadlec@utia.cas.cz to obtain modification details.

- Xilinx Vivado 2015.4 hardware description file (HDF).
- Xilinx SDK workspace with HDMI in HDMI out application.
- Petalinux 2015.4 sources with BSP file prepared for the Vivado 2015.4 design.
- Precompiled SD card files for platform quick test.

2.1 FPGA

FPGA design block diagram is shown in Figure 2, where the main video chain contains HDMI input conversion block (IMAGEON_HDMI_IN), video direct memory access (VDMA) block and HDMI output block (IMAGEON_HDMI_OUT).

The HDMI input block gets video signal with embedded synchronization pulses from HDMI input codec (ADV7611 on IMAGEON extension card), it is repacked to video data (16-bit YCrCb 4:2:2 video format) and separated embedded syncs. After that, this video signal is converted to AXI Stream (AXIS).

Video data mapped on AXIS are then stored frame by frame to frame buffers in DDR3 memory by the VDMA block. VDMA uses 3 frame storages and internal lock between them to avoid image tearing; it always reads or writes the whole frame with constant output frame rate as required by HDMI monitor.

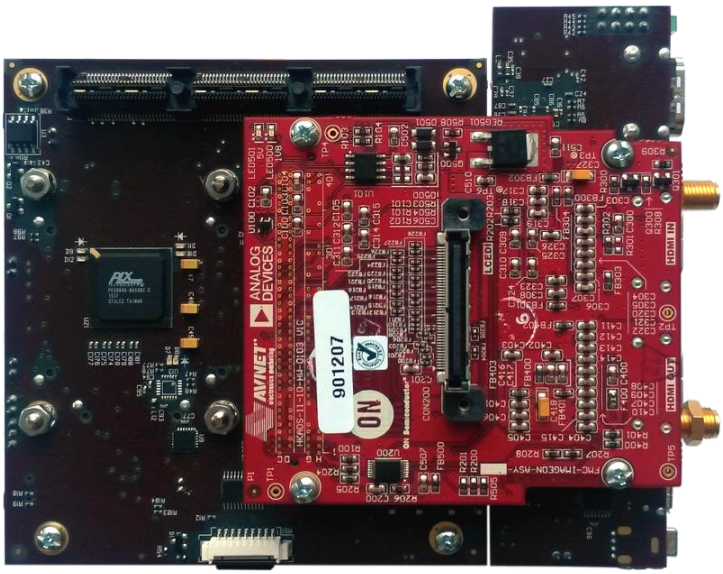

Figure 1: Sundance EMC2-DP v2 carrier board, FPGA SoM Trenz TE0715-7030 (the SoM is located on the bottom side of the carrier) and FMC IMAGEON extension card.

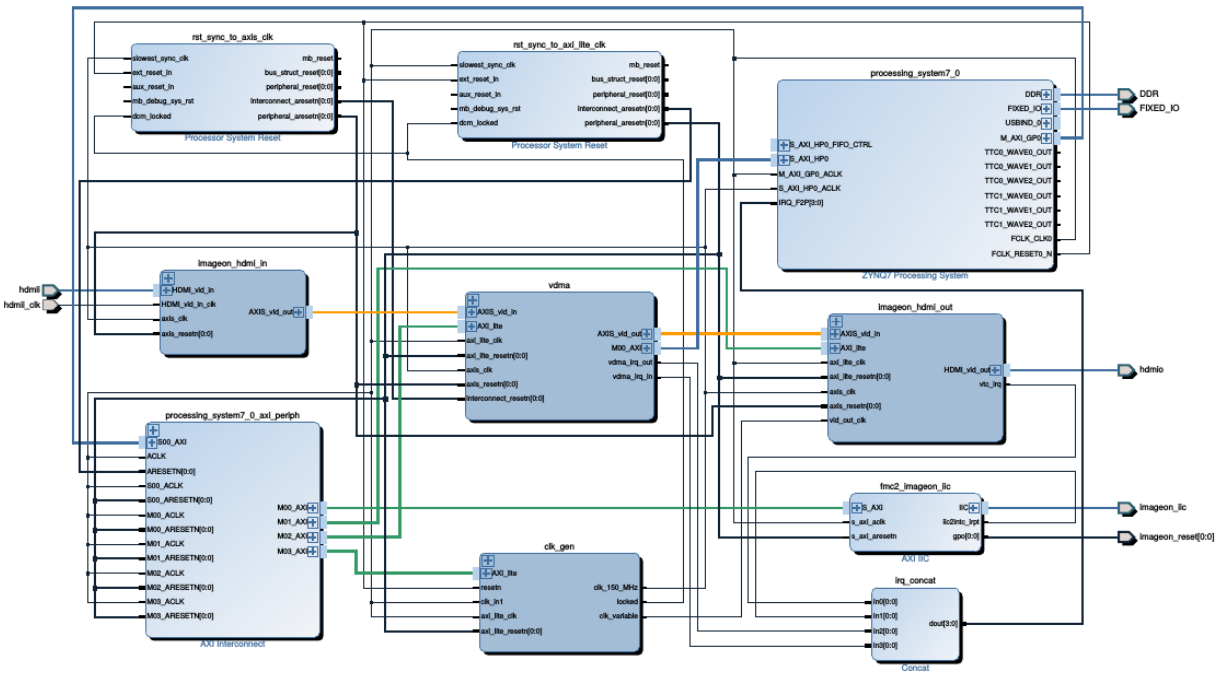

Figure 2: Block diagram of the HW platform.

The HDMI output block gets AXIS with video data from VDMA and converts them back to video signal. The data are synchronized with the timing according to required output resolution and frame rate. This video signal has embedded syncs and it is passed to HDMI output (ADV7511 codec on IMAGEON extension card).

All blocks in programmable logic are configured by an AXI-Lite interface running at 50 MHz. AXIS uses 150 MHz clock to safely transfer full HD (1920x1080) video data at 60 frames per second. The platform also contains programmable clock generator allowing the change of output video resolution at the runtime under ARM SW control.

2.2 Linux

The compilation of Linux image from Petalinux 2015.4 sources is supported by providing platform BSP package file *emc2-plnx.bsp*. This file can be used in a standard way described in Xilinx user guides UG1144, UG1156, UG1157 and UG976 to build Linux image *image.ub* file.

2.3 HDMI in HDMI out Example

This demonstrator provides standalone and Petalinux version of HDMI in HDMI out SW example. Both of them require Xilinx SDK 2015.4 tool to compile and debug them. The example is a simple video pass through demo using 3 frame buffers located in DDR3 memory. The resolution of the HDMI input can be up to full HD (1920x1080) at 60 frames per second. Upper limit is input pixel clock 148.5 MHz (full HD). The HDMI input provides extended display identification data (EDID), Table 1 summarizes supported input video signal resolutions as they are presented by EDID to the input video signal source (usually PC graphic card). The input resolution can be changed on the fly. The serial terminal shows detected video parameters. Output resolution can be changed during the runtime as well. Supported output resolutions are presented in Table 2. If the resolution of the input video signal is less than the output resolution, it will be displayed the input video signal plus black margin to fill the video signal to required output resolution. In case the resolution of the input video signal is greater than the output resolution is required, the output will be created by

cropping of the input video signal. This image window can be moved within the greater resolution range. This behavior is shown in Figure 3.

Table 1: FMC IMAGEON HDMI Input EDID - supported input resolutions.

Resolution	Frame rate
1920x1080	60
1680x1050	60
1600x1200	50
1440x900	60
1366x768	60
1280x1024	60
1280x960	60
1280x800	60
1280x720	60
1152x864	60
1152x720	60
1024x768	60
800x600	60
800x480	60
720x576	60
720x480	60
640x480	60

Table 2: Supported resolutions on FMC IMAGEON HDMI output.

Resolution	Frame rate	Pixel clock [MHz]
1920x1080	60	148.50
1920x1200	50	128.44
1600x1200	50	135.00
1680x1050	60	119.23
1280x1024	60	108.00
1028x720	60	74.25
1024x768	60	65.00
800x600	60	40.00
640x480	60	25.19
600x800	60	40.00

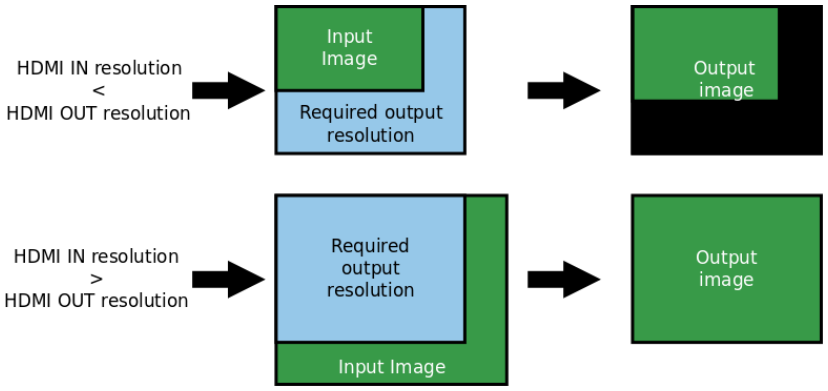

Figure 3: Behavior of the HDMI in HDMI out demo when the input resolution is not the same as the output resolution.

There are a few differences between standalone and Petalinux versions of the SW example. In standalone version, all peripherals are accessed directly using physical addresses. In Petalinux version, they are mapped into the memory user space translating physical addresses to virtual addresses. Another important thing is that the frame buffer location should be outside the program code address space to avoid a risk that another program code will rewrite it. In the standalone version, only one program code is in the main memory. The program address space is well known. The frame buffers can be placed in the address space which is not directly used by the program. In Petalinux, any other running application can allocate memory belonging to the frame buffers. The solution is withdrawing some memory space for the frame buffers from the Linux kernel address space. The memory is reserved for that purpose from address 0x2D000000, size is big enough to hold 3 frames aligned to Kb. The size of this region is 48 MB.

3. Used Tools and Resources

- Sundance EMC^2-DP carrier, Trenz SoM and Avnet FMC extension card.
- 1920x1080p60 capable monitor with HDMI input.
- 2x HDMI cable, micro USB cable, Ethernet cable, power supply.
- Micro SD card formatted with FAT32.
- Petalinux 2015.4 for compilation of Linux image (optional).
- SDK 2015.4 for compilation of example HDMI in HDMI out application.

4. Precompiled Binaries

For the quick run and test of the application, precompiled binaries for standalone and Petalinux version are available in provided package. Examine the package content in Section 6.

4.1 Standalone Application

1. Copy content of *prebuilt/standalone* folder to the root of the micro SD card (*BOOT.bin* file contains all, FPGA bitstream, FSBL and application).
2. Insert the SD card to EMC^2-DP carrier card reader slot.
3. Connect HDMI signal cable from the FMC HDMI output port to the monitor.
4. Connect HDMI signal cable from a video signal source (PC for instance) to the FMC HDMI input port.
5. Connect micro USB cable from the PC to J9 connector on the carrier.
6. Turn on the EMC^2-DP platform.
7. Set up a serial terminal (Putty for instance) with parameters: 115200 bps, 8 bit, 1 stop, no parity, no flow control. The terminal provides a user interface.
8. Observe the terminal for the application output.

   NOTE: The communication between the platform and the terminal cannot be bound before the platform is powered up. As the system boot is fast, it can happen that the application output prints will not be caught by the terminal. To test the application is running press '*m*' key. It prints user menu.

4.2 Petalinux Application

1. Copy content of *prebuilt/petalinux* folder to the root of the micro SD card (*BOOT.bin*, *image.ub* and *hdmii-hdmio-plnx.elf* files).
2. Insert the SD card to EMC^2-DP carrier card reader slot.
3. Connect HDMI signal cable from the FMC HDMI output port to the monitor.
4. Connect HDMI signal cable from a video signal source (PC for instance) to the FMC HDMI input port.

5. Connect micro USB cable from the PC to J9 connector on the carrier.
6. Turn on the EMC^2-DP platform.
7. Set up a serial terminal (Putty for instance) with parameters: 115200 bps, 8 bit, 1 stop, no parity, no flow control. The terminal provides a user interface.
8. Observe the terminal for the application output.

   NOTE: The communication between the platform and the terminal cannot be bound before the platform is powered up. As the system boot is fast, it can happen that the Linux boot sequence will not be caught by the terminal. In this case press *enter* key, it prints a Linux command line.

9. When a Petalinux boot sequence has already finished, start the application from the serial terminal:
   ```
   cd /media/card
   ./hdmii-hdmio-plnx.elf
   ```

5. Compiling the Application from Sources

1. Start Xilinx SDK 2015.4 where set its workspace to *sw* folder.
2. Create a new project with hardware platform specification, chose menu *File→New→Project…→Xilinx→Hardware Platform Specification*, set project name to *hw_platform_0* and browse to HDF file (folder *hw/emc2dp2-30-im-hio/hio.sdk*), see Figure 4.
3. Create a new board support package for standalone version of the application, menu *File→New→Board Support Package*, project name set to *standalone_bsp_0*, set hardware platform to *hw_platform_0*, set CPU to *ps7_cortex9_0* and select *standalone* as board support package OS (Figure 5)
4. Create first stage boot loader (FSBL), menu *File→New→Application Project*, set project name to *fsbl*, select OS platform as *standalone*, set hardware platform to *hw_platform_0*, set CPU to *ps7_cortex9_0*, check *create new* in board support package selection, let BSP name *fsbl_bsp*. Click on N*ext* button. Select *Zynq FSBL* template and click *Finish* (Figure 6).

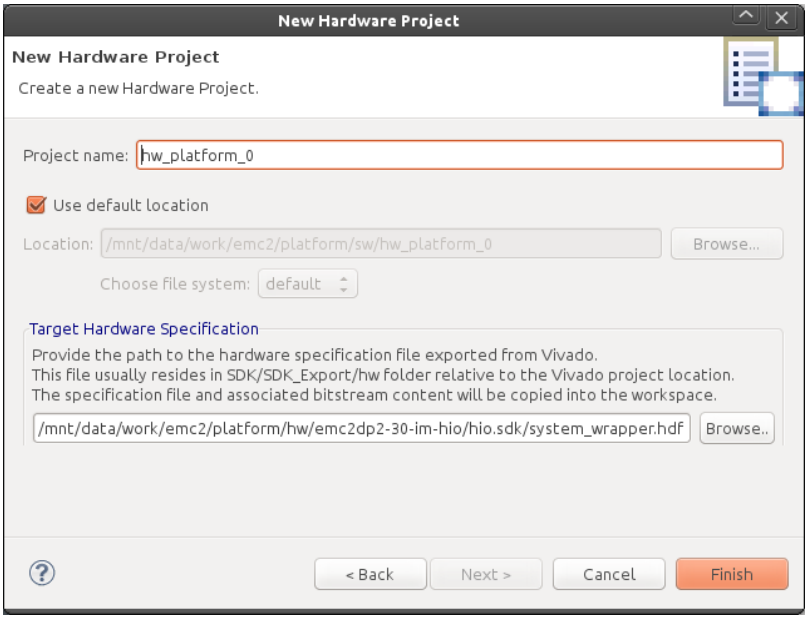

Figure 4: Create a new hardware platform specification project.

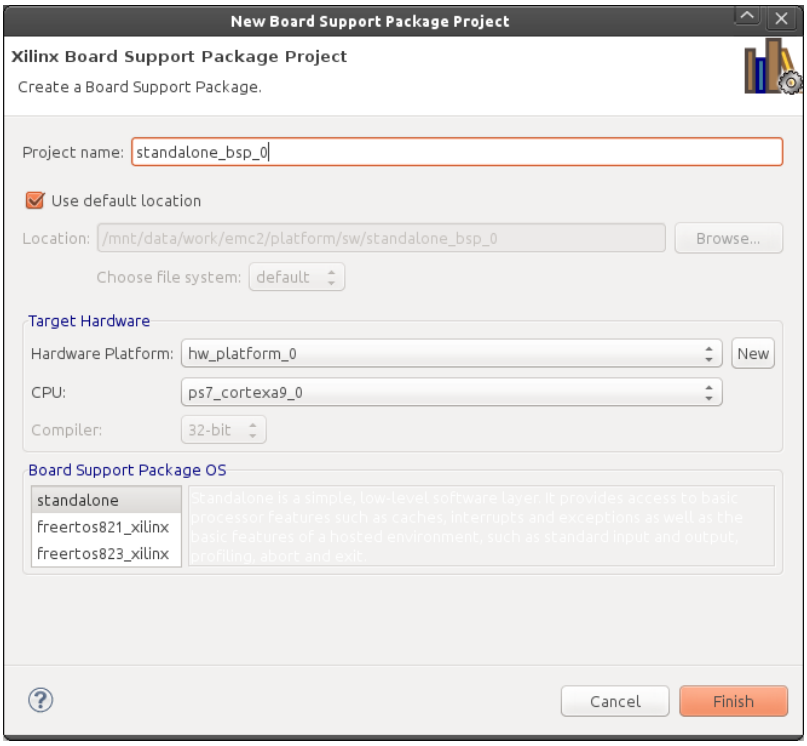

Figure 5: Create a new standalone board support package.

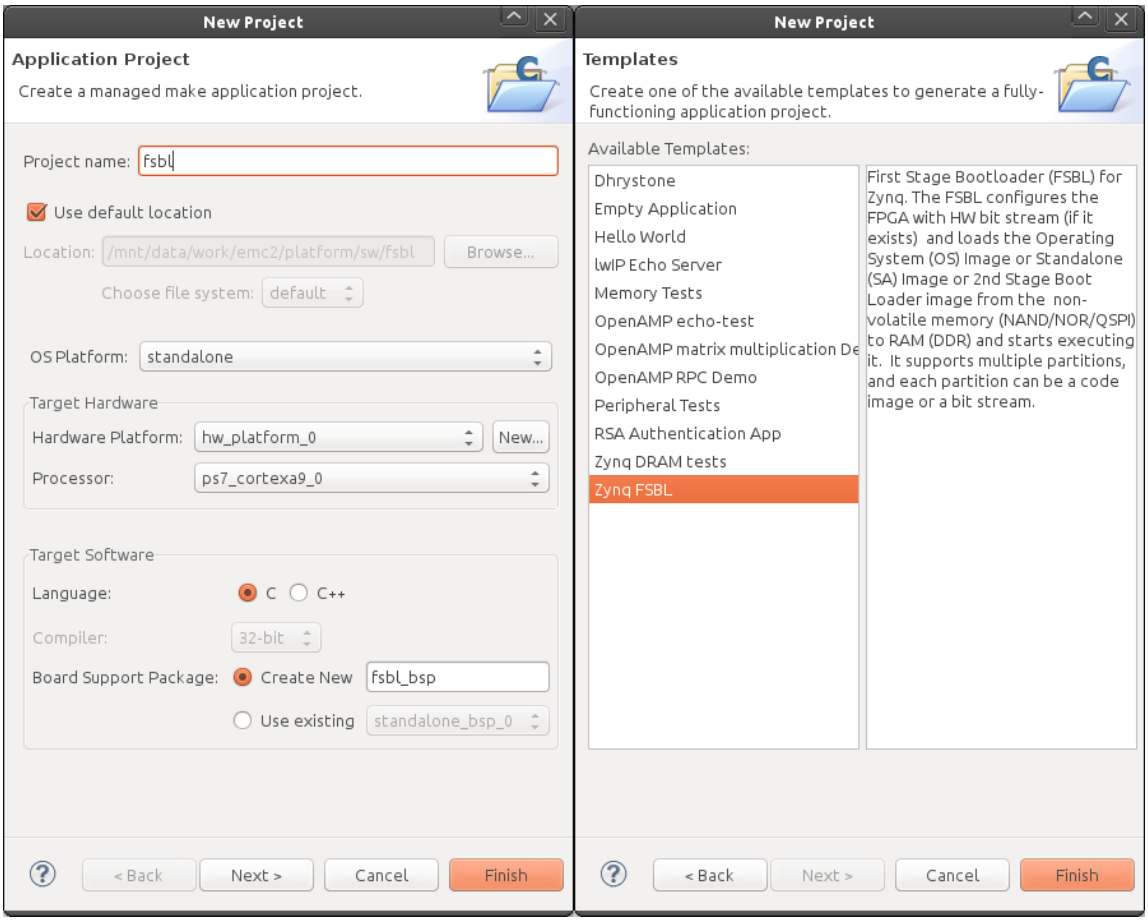

Figure 6: Create FSBL.

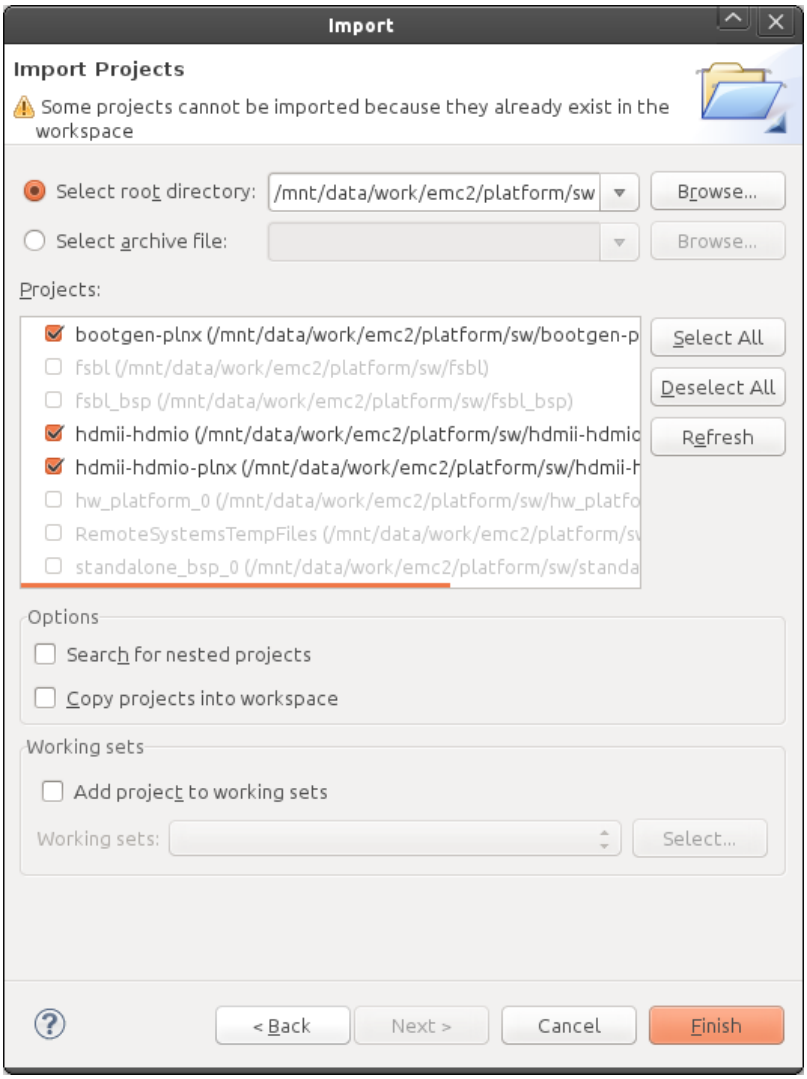

Figure 7: Import SW project into the SDK workspace.

5. Import prepared application projects into the SDK workspace, select menu *File→Import…→General→Existing Projects into Workspace*, browse to folder *sw* and select projects (Figure 7):
   - *bootgen-plnx* – generates *BOOT.bin* file for Petalinux,
   - *hdmii-hdmio* – standalone application,
   - *hdmii-hdmio-plnx* – Petalinux application.

   All imported projects should be automatically compiled. If they are not, rebuild them.
6. Generate *BOOT.bin* file for standalone application. In the project explorer, select standalone project *hdmii-hdmio*, right click on it, select *Create Boot Image* from the context menu and click *Create Image* button.

7. Copy new files to micro SD card.

Standalone

File	Location	Description
BOOT.bin	sw/hdmii-hdmio/bootimage	Includes FPGA bitstream, FSBL and application

Petalinux

File	Location	Description
BOOT.bin	sw/bootgen-plnx/bootimage	Includes FPGA bitstream, FSBL and u-boot
image.ub	prebuilt/petalinux	Linux image
hdmii-hdmio-plnx.elf	sw/hdmii-hdmio-plnx/Debug	application

6. Package Contents

```
.
|- doc/
|    `- emc2-hio-appnote-v2.pdf
|- hw/
|    `- emc2dp2-30-im-hio/
|- petalinux/
|    `- emc2-plnx.bsp
|- prebuilt/
|    |- petalinux/
|    `- standalone/
`- sw
     |- bootgen-plnx/
     |- hdmii-hdmio/
     `- hdmii-hdmio-plnx/
```

7. References

[1] Sundance.Technology, „EMC²-DP,“ 11 07 2016. [Online]. Available: http://www.sundance.technology/som-cariers/pc104-boards/emc2-dp/

[2] Trenz Electronic, „Trenz Electronic TE0715 Series (Z-7015, Z-7030),“ 11 07 2016. [Online]. Available: http://www.trenz-electronic.de/products/fpga-boards/trenz-electronic/te0715-zynq.html

[3] AVNET, „HDMI Input/Output FMC Module,“ 14 07 2016. [Online]. Available: https://products.avnet.com/shop/en/ema/kits-and-tools/development-kits/3074457345628965509

8. Disclaimer

This disclaimer is not a license and does not grant any rights to the materials distributed herewith. Except as otherwise provided in a valid license issued to you by UTIA AV CR v.v.i., and to the maximum extent permitted by applicable law:

(1) THIS APPLICATION NOTE AND RELATED MATERIALS LISTED IN THIS PACKAGE CONTENT ARE MADE AVAILABLE "AS IS" AND WITH ALL FAULTS, AND UTIA AV CR V.V.I. HEREBY DISCLAIMS ALL WARRANTIES AND CONDITIONS, EXPRESS, IMPLIED, OR STATUTORY, INCLUDING BUT NOT LIMITED TO WARRANTIES OF MERCHANTABILITY, NON-INFRINGEMENT, OR FITNESS FOR ANY PARTICULAR PURPOSE; and

(2) UTIA AV CR v.v.i. shall not be liable (whether in contract or tort, including negligence, or under any other theory of liability) for any loss or damage of any kind or nature related to, arising under or in connection with these materials, including for any direct, or any indirect, special, incidental, or consequential loss or damage (including loss of data, profits, goodwill, or any type of loss or damage suffered as a result of any action brought by a third party) even if such damage or loss was reasonably foreseeable or UTIA AV CR v.v.i. had been advised of the possibility of the same.
Critical Applications:

UTIA AV CR v.v.i. products are not designed or intended to be fail-safe, or for use in any application requiring fail-safe performance, such as life-support or safety devices or systems, Class III medical devices, nuclear facilities, applications related to the deployment of airbags, or any other applications that could lead to death, personal injury, or severe property or environmental damage (individually and collectively, "Critical Applications"). Customer assumes the sole risk and liability of any use of UTIA AV CR v.v.i. products in Critical Applications, subject only to applicable laws and regulations governing limitations on product liability.