**OV5640 video for linux based on ZYBO——User guide**

**Summery:**

**This design uses ov5640 which is a low voltage, high-performance, 1/4-inch 5 megapixel CMOS image sensor and provides a whole HD video solution.**

**Note :**

* Vivado：2015.2
* Divecetree：2015.1
* Kernel：2014.4
* Rootfs : Linaro
* U-boot : 2015.01
* Board : ZYBO

**Setting up environment**

* This design uses a self-defined bus named cap\_dvp\_io\_v1\_0. So if you want to implement this design, add this interface into vivado 2015.2. Add <your path>/vivado\_bus/cap\_dvp\_io\_v1\_0 to<your path>/Xilinx\Vivado\2015.2\data\ip\interfaces.and replace<you path>/Xilinx\Vivado\2015.2\data\ip\vv\_index with <your path>/vivado\_bus/vv\_index.
* Finally, you can use vivado to run the project tcl.

**Procedures**

**Building hardware project**

* Download the zip and extract it
* Open vivado 2015.2 tcl console, then go to <your path>/hardware\_prj
* Run source ./system\_prj.tcl
* Waitting for bitstream ... then it's ok !
* Note: The ip\_repo folder also contains a few existing video processing ipcores,such asaverage filter,sobel filter,morph operator,threshold,grayand so on. These ipcores are flexiable and can provide extra functionalities. In order to use these functions, just add them to the block-diagram.

**Creating linux img**

* Using the generated bitstream to create the BOOT.bin.(u-boot,fsbl just as usual).
* Using the SDK to create the Devicetree. There are a few modifications on the source code. Please refer to folder"dts" for detailed information. PS: There is a bug on 2015.1 devicetree: fclk-enable = <0x0>; Plese modify it to fclk-enable = <0xf>;
* Creating the uImage: copy <your path>/v4l2\_kernel\_2014.4\_patch/v4l2.patch to kernel sources folderlinux-xlnx and patch it. Then select a few options.
* Here is the list:
  + --- V4L platform devices   
    < > Marvell 88ALP01 (Cafe) CMOS Camera Controller support   
    < > SoC camera support   
    <M> Xilinx Video IP (EXPERIMENTAL)   
    < > Xilinx Video Color Filter Array   
    < > Xilinx Video Chroma Resampler   
    < > Xilinx Video HLS Core   
    < > Xilinx Video Remapper   
    < > Xilinx Video RGB to YUV Convertor   
    <M> Xilinx Video Scaler   
    < > Xilinx Video Switch   
    < > Xilinx Video Test Pattern Generator   
    < > Xilinx Video Timing Controller
  + --- DMA Engine support   
    [ ] DMA Engine debugging   
    \*\*\* DMA Devices \*\*\*   
    [ ] ARM PrimeCell PL080 or PL081 support   
    < > Synopsys DesignWare AHB DMA support   
    < > Synopsys DesignWare AHB DMA platform driver   
    < > Synopsys DesignWare AHB DMA PCI driver   
    <\*> DMA API Driver for PL330   
    < > Freescale eDMA engine support   
    [\*] Xilinx DMA Engines   
    <M> Xilinx AXI VDMA Engine   
    < > Renesas Type-AXI NBPF DMA support   
    \*\*\* DMA Clients \*\*\*   
    [ ] Async*tx: Offload support for the async*tx api   
    < > DMA Test client
* Compiling the kernel and driver modules. Then copy thevideobuf2-core.ko, videobuf2-dma-contig.ko,videobuf2-memops.ko,xilinx\_vdma.ko,xilinx-axi-video.ko ,xilinx-scaler.koto a new folder named 'drivers'

**Implementation**

* Copy BOOT.bin, uImage, devicetree.dtb and folders drivers,<your path>/cvapp,ov5640\_init to SD card.
* Power up the board.
  + Cd ov5640\_init, Run ./ov5640 to init the camera.
  + Insmod all drivers which in drivers.
  + Cd <your path>/cvapp, run python pic\_capture.py, then you can get a bmp file. You can see a picture captured by the camera when you open it.

Congratulations!

**Best Regards!**