

Demonstration Tools on High Performance FPGA

Industry-Based Vision Processing Application and FPGAs:

This report is about the final design of application which demonstrates the efficiency of high performance FPGA. In this design, we have employed vision processing application on FPGA. As manufacturing industry is critically dependent upon human beings' ability to analyse complex visual scenes. However, as human are prone to errors. This critical tasks can be efficiently managed if the computer vision techniques are employed. It is an established fact that the computer vision is potentially able to fulfil the needs of a very diverse range of industrial inspection, monitoring and control tasks. However, in modern industrial application ``Embedded Vision'' are most relevant rather than traditional machine vision systems. Embedded vision systems need to be highly compact and function in highly challenging and unstructured environments, while still delivering high quality images.

It is worth mentioning that embedded vision is still an emerging technology, to date there are typically two main types of processors used in embedded systems – field programmable gate arrays (FPGAs) and graphics processing units (GPUs). In recent years FPGAs are gaining favour as an embedded vision processor than GPUs or general purpose processors. FPGAs are much faster than general processors and thus, they have been gaining popularity because of their extremely low latency levels. They also provide significantly more processing potential with far lower energy consumption, and they can accelerate several portions of a computer vision pipeline, where GPUs can only accelerate one.

High Performance Vision Processing Using PYNQ FPGA:

Architecture:

Xilinx new PYNQ FPGA is based on ZYNQ architecture. first SoC released by Xilinx. ZYNQ devices comprise a *Processing System (PS)*, and *Programmable Logic (PL)*, the PL being equivalent to that of a Field Programmable Gate Array (FPGA) and the PS side contains high performance cortex A9 ARM processor as shown in figure 1.

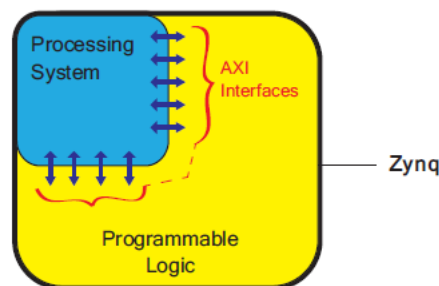


Figure 1: Simplified model of Zynq Architecture

In this project, we are using PYNQ Board as shown in figure 2.

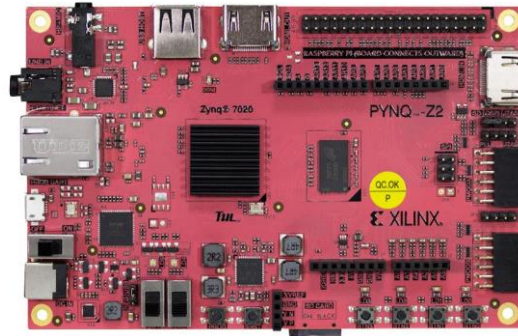


Figure 2: PYNQ Board

PYNQ FPGA provides framework for creating and using applications using python programming. It is based on Zynq FPGA. The name 'PYNQ' is derived from 'Python productivity for Zynq'.

PYNQ framework in three layers. The bottom layer represents the basic hardware design. This normally created in Vivado using IP Integrator and associated design tools, and then generated to a bitstream (*.bit) file. The mid-layers of the PYNQ consist of Python software, and the OS and low level software drivers which can access the low level hardware. At the top level, user interaction is facilitated by python development framework like Jupyter. This framework is shown in figure 3.

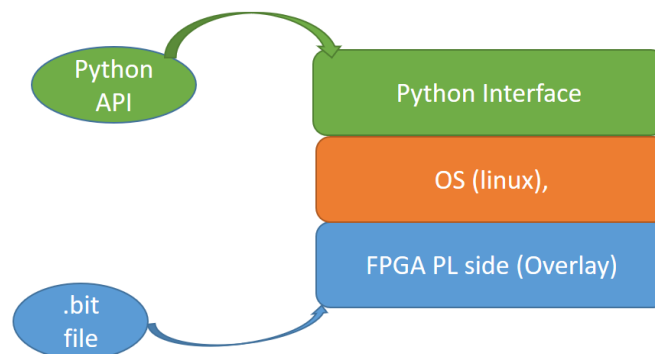


Figure 3: PYNQ Framework

Real-time Video Processing:

Real-time video and image processing is used in a wide variety of applications from video surveillance and traffic management to medical imaging applications. These operations typically require very high computation power. However, the employed PYNQ provides the necessary performance for real-time image and video processing.

Edge detection is a fundamental tool used in most image processing applications to obtain information from the frames as a precursor step to feature extraction and object segmentation. This process detects outlines of an object and boundaries between objects and the background in the image. An edge-detection filter can also be used to improve the appearance of blurred or anti-aliased video streams. The basic edge-detection operator is a matrix- area gradient operation that determines the level of variance between different pixels. Thus, it demands intensive computation.

Possible Industrial Application:

This image acquisition and processing system will assist to measure the width and inspect the quality of the stainless steel strip in a production line. Specifically, Stainless steel is initially produced in the melting shop, and in the subsequent hot rolling mill it is usually shaped as strip coils several hundred-meter-long and different width and thickness. The strip width and thickness must be monitored as the dimensions are important features to fulfil the customer orders. This application scenario is discussed in [1].

PYNQ-Based Hardware Design:

In PYNQ, hardware system designs are referred to as *overlays*. They can be used in a manner analogous to software libraries. Specifically, overlay represents complete hardware system that will be programmed onto the PL (FPGA side), and it represents part of the hardware / bottom layer of the PYNQ framework. In this design we have used two overlays:

1. The Base Overlays for basic I/O communication and video interface.
2. The Computer Vision overlays to carry out video processing.

This overlays are created using Vivado HLS software. The CV overlay is shown in figure 4.

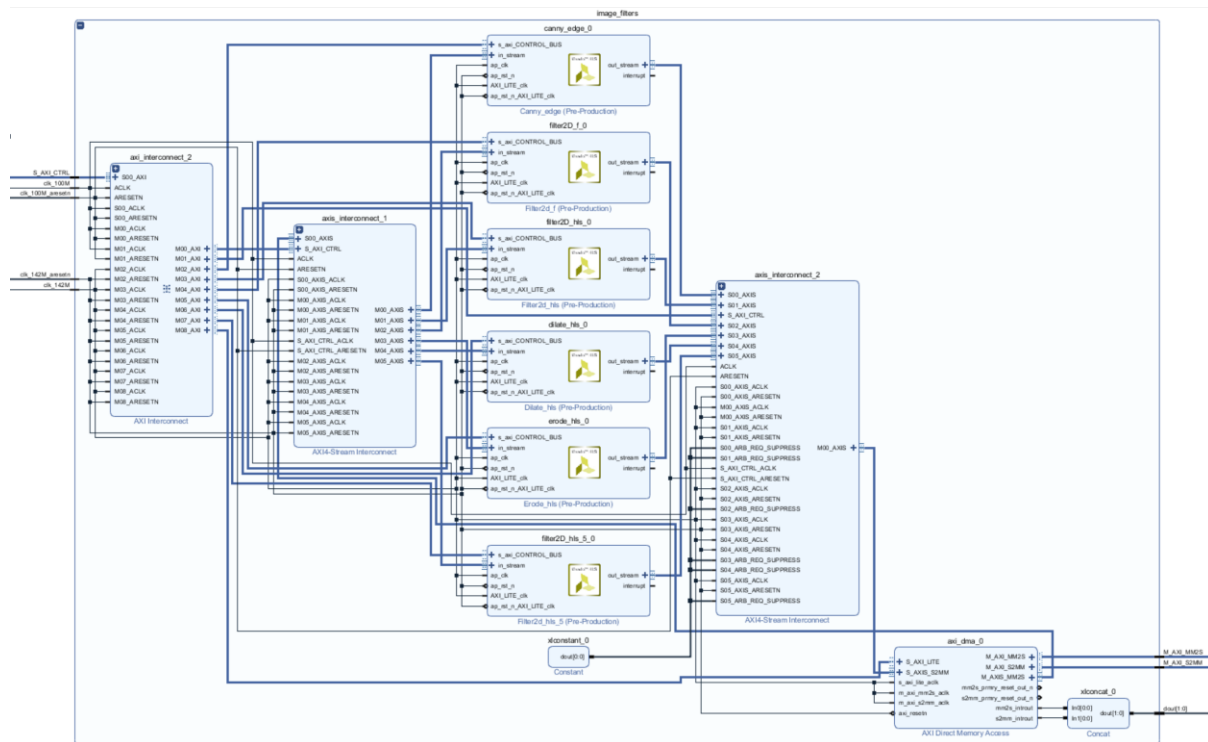


Figure 4: PYNQ Hardware Design (Compute Vision Overlay)

Each vivado block of this design (as shown in Figure 5) represents each computer vision functions. This functions are written in high level language and corresponding hardware version has been generated using Vivado HLS.



Figure 5: Core created using Vivado HLS for Dialate operation

PYNQ-Based Software Design:

As a software development, the video processing is carried out using python. It has to be noted that the Python coding is not used for hardware description and verification, i.e. to generate circuits for implementation in the PL section. Python is used for programming on the PS only, which includes interaction with hardware via the PS-PL interfaces. This means that the python interface executed on PS side and it Communicates with PL. In the PL side, computation intensive processing is carried out. This is shown in figure 6.

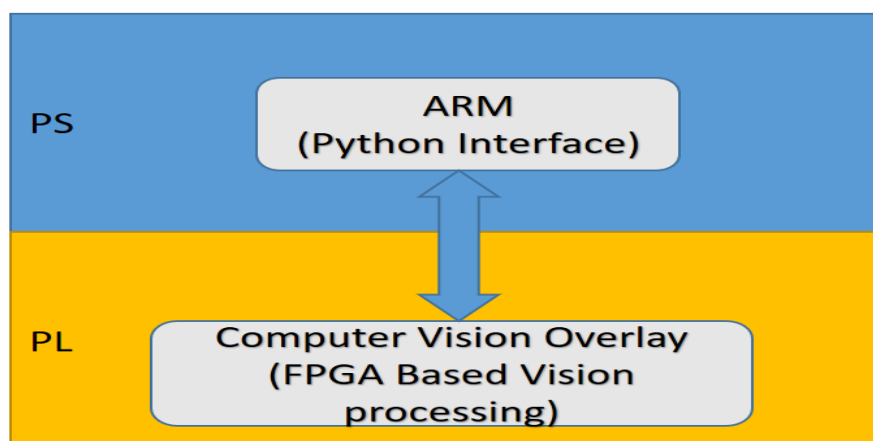


Figure 6: PS & PL Communication

Setup:

In this setup, we are feeding HDMI video to the PYNQ FPGA and the processed image is shown in the monitor. Communication between the board and the PC has been carried out through Ethernet communication. In this work, we are detecting the edge of the video frame. As it can be seen that the through the output stream, we can detect the lane. The design set up at University of Essex, is shown in figure 7.

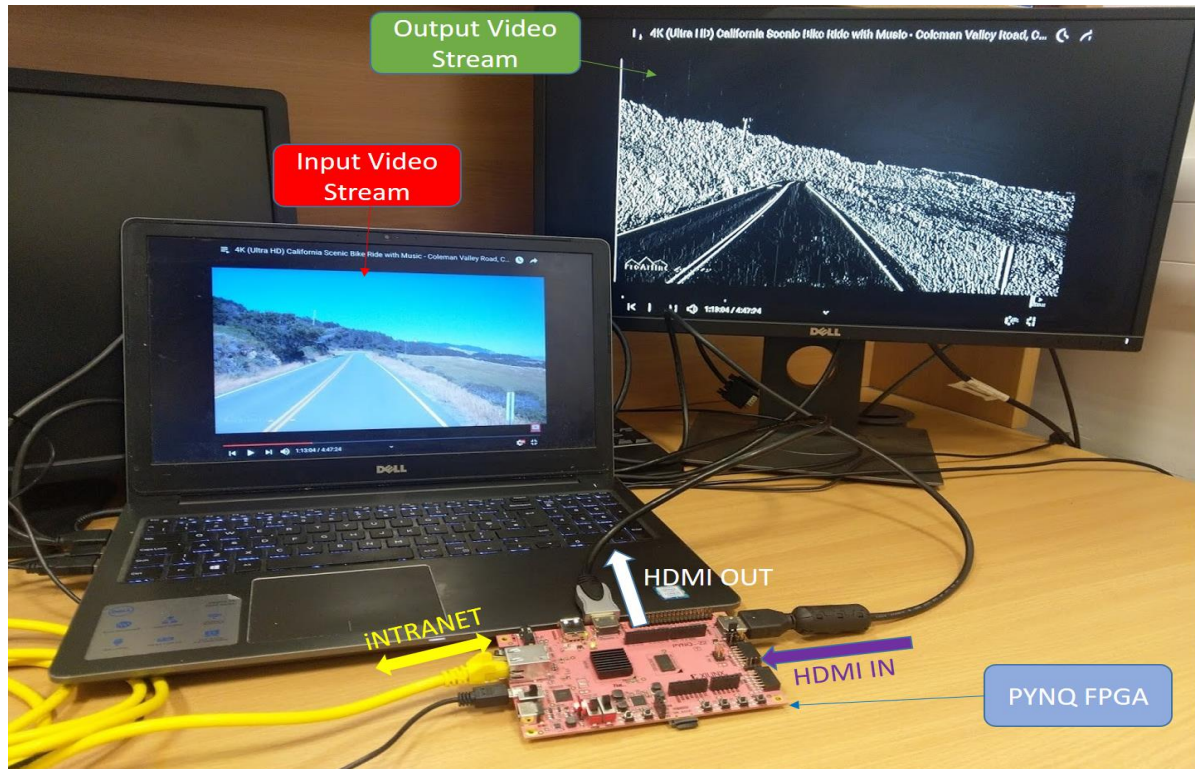
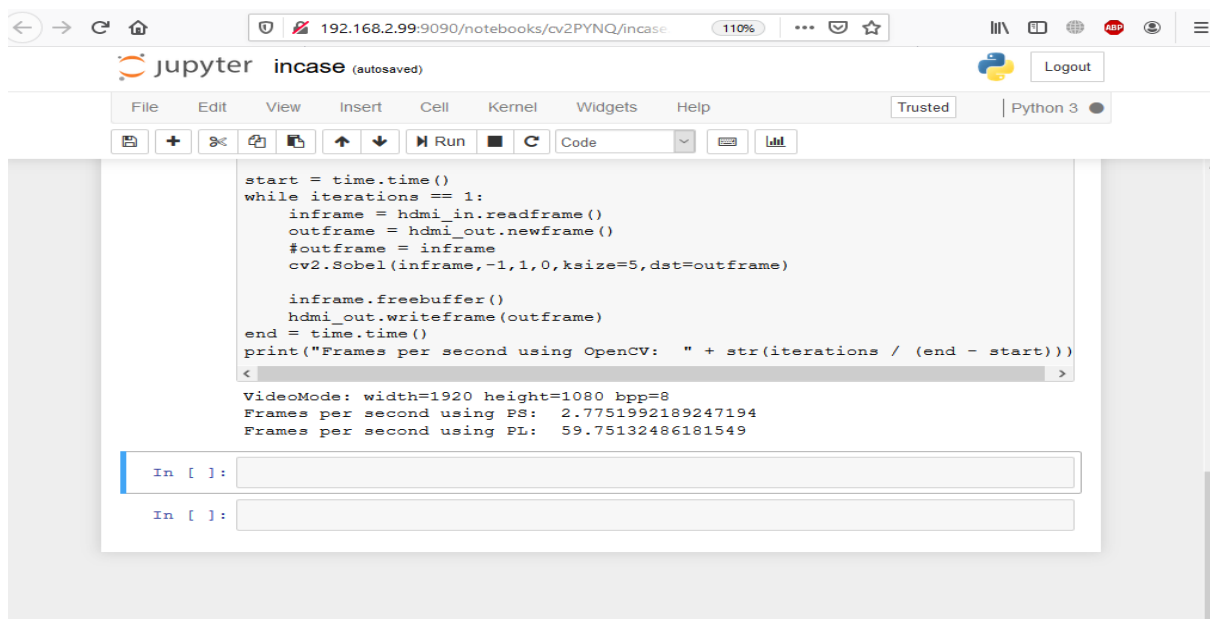


Figure 7: Setup @ University of Essex



```

start = time.time()
while iterations == 1:
    inframe = hdmi_in.readframe()
    outframe = hdmi_out.newframe()
    #outframe = inframe
    cv2.Sobel(inframe, -1, 1, 0, ksize=5, dst=outframe)

    inframe.freebuffer()
    hdmi_out.writeframe(outframe)
end = time.time()
print("Frames per second using OpenCV: " + str(iterations / (end - start)))

```

VideoMode: width=1920 height=1080 bpp=8
Frames per second using PS: 2.7751992189247194
Frames per second using PL: 59.75132486181549

In []:

In []:

Figure 8: FPS Observed from PS and PL execution

Results:

We have measured the efficiency of the approach by executing the application in the PS side i.e. completely software based execution and the by executing with FPGA. In order to measure the acceleration, we have measured the number of frames that can be processed using software based solution and using FPGA based solution. It is shown in figure 9. From the figure it can be observed that PS can achieve nearly 3 FPS while the PL side (the FPGA-based solution) is much faster and can achieve the FPS value of 60 (as shown in figure 8). Hence, there is 20x speed.

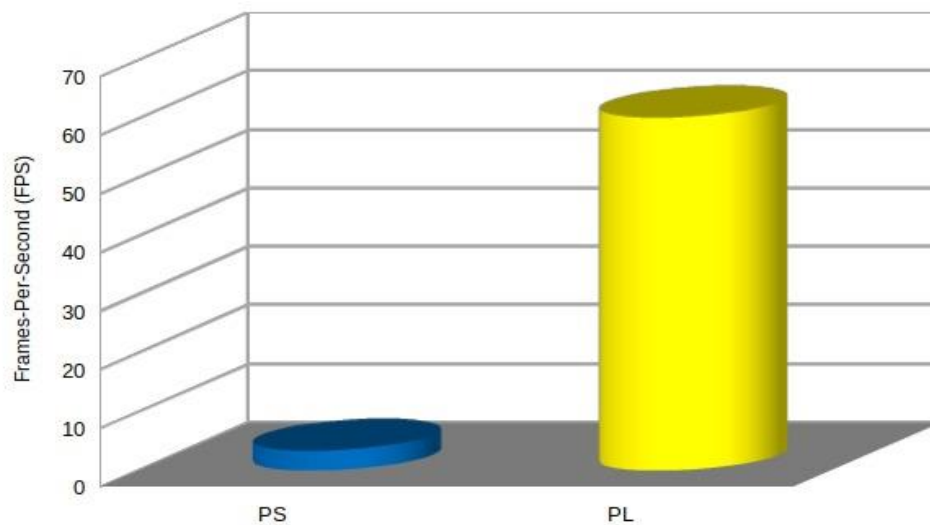


Figure 8: Performance Speed up using FPGAs

Conclusion:

In this report, we have discussed regarding the demonstration tools of high performance FPGA. We have employed real-time video processing technique as our application. Through the comprehensive design setup, it has been verified that FPGA performs faster than software based execution.

Reference:

- [1] Spinola CG, Canero J, Moreno-Aranda G, Bonelo JM, Martin-Vazquez M. Real-time image processing for edge inspection and defect detection in stainless steel production lines. In 2011 IEEE International Conference on Imaging Systems and Techniques 2011 May 17 (pp. 170-175). IEEE.
- [2] Crockett, Louise, David Northcote, Craig Ramsay, Fraser Robinson, and Robert Stewart. "Exploring Zynq MPSoC: With PYNQ and Machine Learning Applications." (2019).