3. PYNQ FRAMEWORK FOR FPGA-BASED EMBEDDED SYSTEMS

Marmuzevich M.A., Master Degree Student gr.255741

Belarusian State University of Informatics and Radioelectronics
Minsk, Republic of Belarus

Liakh Y. V. – Senior Lecturer

Annotation. In the article the advantages and disadvantages of PYNQ technology for embedded design are considered. Also, the capabilities and scope of this technology are presented. This approach is compared to other technologies in this field.

Keywords. Technology, embedded systems, Field Programmable Gate Array, computer vision, object detection, neural networks, programming language.

There are some problems of embedded systems development such as power consumption, dimension and weight restrictions and increased requirements for system performance. This forces engineers and scientists to resort to the creation of specialized systems and reduce the use of universal microcontrollers and GPUs (Graphics Processing Units). The solution to most problems is increasingly becoming FPGA (Field Programmable Gate Array) systems. Despite the advantages of this technology, there is a problem of the complexity of developing process. The main languages for FPGA programming are VHDL and Verilog HDL [1], which are very specific and complex for software programmers to study. However, AMD company has created a framework called PYNQ [2].

What is PYNQ? PYNQ is an open-source project that makes it easier to use adaptive computing platforms. Using the Python language and libraries, designers can exploit the benefits of programmable logic and microprocessors to build more capable and exciting electronic systems. PYNQ can be used with Zynq, Zynq Ultrascale+, Zynq RFSoC, Alveo accelerator boards and AWS-F1 to create high performance applications with parallel hardware execution; high frame–rate video processing; hardware accelerated algorithms; real–time signal processing; high bandwidth IO; low latency control.

This technology is intended to be used by a wide range of designers and developers including:

– software developers who want to take advantage of the capabilities of Adaptive Computing platforms without having to use ASIC-style design tools to develop hardware;

– system architects who want an easy software interface and framework for rapid prototyping and development of their Zynq, Alveo and AWS–F1 design;

– hardware designers who want their designs to be used by the widest possible audience.

This technology is widely used in the field of object detection and video processing. PYNQ includes a big library of IP blocks for audio and video processing, interaction with the periphery, operation with data and other specific blocks. Consider the problem of constructing a video stream. The interface part is an HDMI block, with the ability to adjust the image resolution by configuring the corresponding modules. An example from the standard library PYNQ.lib HDMI-video pipeline was used to test the performance. Python code:

```
from pynq.overlays.base import BaseOverlay
from pynq.lib.video import *

base = BaseOverlay("base.bit")
hdmi_in = base.video.hdmi_in
hdmi_out = base.video.hdmi_out
hdmi_in.configure()
hdmi_out.configure(hdmi_in.mode)

hdmi_in.start()
hdmi_out.start()
hdmi_in.tie(hdmi_out)
```

As for the implementation in Vivado, the operation is similar. It is possible to process an image in a video stream in Python by using the OpenCV library [3]. The OpenCV library is an open-source library that contains algorithms for solving problems in computer vision, image processing, and general-purpose numerical algorithms. It is written in C and C++ and can run on computers with different operating systems. Its main goal is to provide a simple interface that allows you to develop applications using the capabilities of computer vision. The OpenCV library contains many functions from different areas of computer vision.

OpenCV is structured around four main components (Figure 1). The first component contains the main algorithms for image processing and high-level machine vision algorithms. The next component is a machine learning library that can solve statistical classification and clustering problems. There is also a component that provides functions for interacting with the operating system, file system, and computer hardware. It allows you to display various images, work with cameras to record videos or photos, save information to files. Another component simply stores the basic data structures for working with other modules.

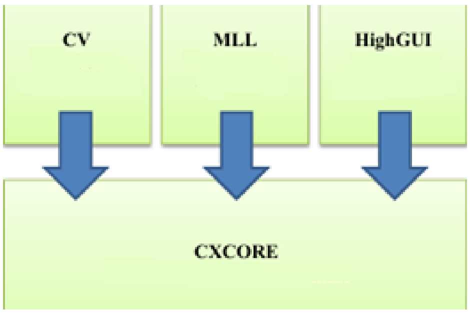

Figure 1 – The main modules of the OpenCV library

But when solving problems using this library, it is necessary to take into account its pros and cons. The advantages include active community and library support; free access; abundance of built-in algorithms; high speed of algorithm execution in comparison with MatLab. The disadvantages are lack of error handling codes; focus on large platforms, poor adaptability to embedded solutions.

Further, after receiving the image and testing the performance it is necessary to implement a similar interface using IP blocks in the Vivado development environment in order to reduce the consumed resources and better differentiate the video stream into blocks.

It is necessary to consider the video stream block by block. To clock the operation of the entire system, the PixelClk signal generated by the block for converting the DVI interface, on the basis of which HDMI is built, into RGB format is used. The reference clock signal for further generation is obtained by converting sys_clock with a frequency of 125 MHz to 200 MHz using the Clocking Wizard frequency multiplier block. The structure of the input channel is shown in Figure 2.

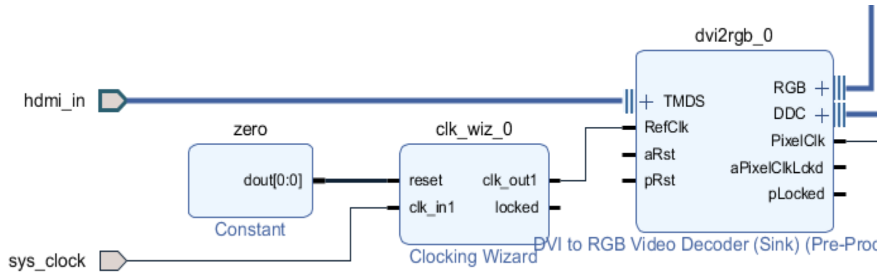

Figure 2 – Input channel of the video stream

When processing a signal in RGB format, it is possible to avoid further conversions and directly output the image through the RGB2DVI block. The block is shown in Figure 3.

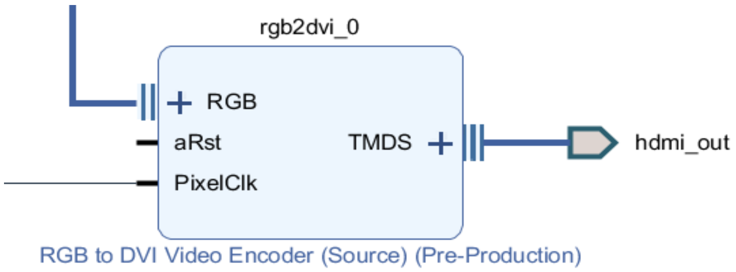

Figure 3 – Output channel of the video stream

The resulting solution allows you to pass a video image with a minimum delay, however, the built–in Xilinx tools allow you to process signals using other interfaces.

For further processing, the video must be cropped into separate pictures, which can be freely used for processing. To store images, a VDMA block is used; it is connected to the video stream via the AXI (Advanced extensible interface). To connect the VDMA unit, it is necessary to use the RGB2AXI, AXI2RGB interface converters, as well as the Video Timing Controller, to synchronize the image output when using additional processing. The block diagram of this solution is shown in Figure 4.

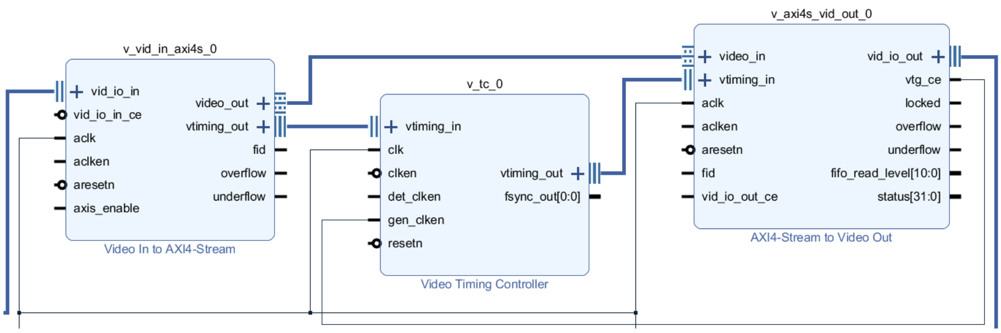

Figure 4 – Video signal preprocessing block

One more example of using this technology is the porting of Binarized neural network on FPGA Zynq-7020. Zynq-Z2 development board will be used for an experiment [4]. The neural network, used in the experiment, was developed by the Norwegian University of Science and Technology [5] using the Cifar-10 dataset; a peculiarity of using a neural network is processing of JPG images. The source image for determining the object is presented in Figure 5.

Figure 5 – Image to recognition

The results of comparing software execution and hardware implementation are presented in Table 1. The results show the obvious superiority of specialized systems over universal ones. However, the implementation of this algorithm is not difficult due to the use of the PYNQ library.

Comparing this technology with commonly used solutions, the closest one is the Arduino platform [6]. Of course, it cannot be used in difficult projects where engineers fight for each clock tact and perform operations as fast as hardware can. However, this platform gives the society such things as smart home things, weather stations, tabloids and millions of devices created around the world and transformed into the industrial form.

Table 1 – Comparison of the speed of software and hardware implementation of Binarized neural network

	Hardware implementation	Software implementation
Time of processing, ms	1583	1588409
Classification frequency, frame/s	631,71	0,63
Class number	4	4
Class name	Deer	Deer

To sum up, PYNQ gives many opportunities for the development of more and more complex systems and their community increases every day. There are a lot of basic examples, that can be downloaded and improved for each task, and this fact can significantly reduce development time and, as a result, the cost of the final product.

References:

1. VHDL vs. Verilog [Electronic resource]. – Mode of access: https://nandland.com/lesson–16–vhdl–vs–verilog–which–language–should–you–learn–first/. – Date of access: 03.03.2023.

2. What is PYNQ? [Electronic resource]. – Mode of access: pynq.io. – Date of access: 02.03.2023.

3. Using OpenCV library for face recognition [Electronic resource]. – Mode of access: https://eduherald.ru/ru/article/view?id=19602/. – Date of access: 12.03.2023.

4. A Framework for Fast, Scalable Binarized Neural Network Inference / ed. Y. Umuroglu, N. J. Fraser. – Boulder : Norwegian University of Science and Technology, 2016, 596 p.

5. Xilinx support portal [Electronic resource]. – Mode of access: https://www.xilinx.com/support/university/xup–boards/XUPPYNQ–Z2.html. – Date of access: 05.03.2023.

6. What is Arduino? [Electronic resource]. – Mode of access: https://www.arduino.cc/. – Date of access: 05.03.2023.