PhD Forum Towards embedded heterogeneous FPGA-GPU smart camera architectures for CNN inference

W. Carballo-Hernández, F. Berry, Maxime Pelcat, M. Arias-Estrada

PhD Forum Towards embedded heterogeneous FPGA-GPU smart camera architectures for CNN inference

W. Carballo-Hernández, F. Berry, Maxime Pelcat, M. Arias-Estrada

PhD Forum: Towards Embedded Heterogeneous FPGA-GPU Smart Camera Architectures for CNN Inference

Walther Carballo-Hernández[1], François Berry[2]
Department of Images, Perception Systems and Robotics
Institut Pascal
Aubière, France
[1]walther.carballo_hernandez@uca.fr
[2]francois.berry@uca.fr

Maxime Pelcat
Department of Images
Institut National des Sciences Appliquées (INSA) des Rennes, IETR, UMR CNRS
Rennes, France
mpelcat@insa-rennes.fr

Miguel Arias-Estrada
Department of Computer Science
Instituto Nacional de Astrofísica, Óptica y Electrónica (INAOE)
Puebla, Mexico
ariasmo@inaoep.mx

Abstract

The success of Deep Learning (*DL*) algorithms in computer vision tasks have created an on-going demand of dedicated hardware architectures that could keep up with the their required computation and memory complexities. This task is particularly challenging when embedded smart camera platforms have constrained resources such as power consumption, Processing Element (*PE*) and communication. This article describes a heterogeneous system embedding an FPGA and a GPU for executing CNN inference for computer vision applications. The built system addresses some challenges of embedded CNN such as task and data partitioning, and workload balancing. The selected heterogeneous platform embeds an Nvidia® Jetson TX2 for the *CPU-GPU* side and an Intel Altera® Cyclone10GX for the *FPGA* side interconnected by *PCIe* Gen2 with a *MIPI-CSI* camera for prototyping. This test environment will be used as a support for future work on a methodology for optimized model partitioning.

Categories and Subject Descriptors Processor Architectures [*Other architectures styles*]: [Neural nets]; Processor Architectures [*Other architectures styles*]: [Heterogeneous (hybrid) systems]; Embedded and Cyber-Physical Systems [*Embedded systems*]: [Embedded hardware]; Integrated Circuits [*Reconfigurable logic and FPGAs*]: [Hardware accelerators]; Architectures [*Parallel architectures*]: [Single instruction, multiple data]

General Terms Artificial Neural Networks (*ANN*), Deep Learning (*DL*), Convolutional Neural Networks (*CNN*), Field Programmable Gate Array (*FPGA*), Graphic Processing Unit (*GPU*), Processing Elements (*PE*)

Keywords Heterogeneous Computing, Edge Computing, Internet of Things, Parallel Programming, Single Instruction Multiple Data, Pipelining, Models of Computation and Architecture

1. Introduction

Deep Learning techniques have become in the last decade the defacto choice for multiple domains, achieving a performance similar to that of a human or even outperforming it in popular and well known competitions. During this period of tremendous evolution with many ground-breaking modifications, it has been observed high accuracy in tasks such as classification, object tracking, feature selection or detection, segmentation, input generation or input reconstruction in multiple domains like: natural language processing and in vision domains, such as image processing and video analytics. Heterogeneous computing comprises sequential and parallel mapping of a certain application or subtask to the best available individual device on a system. It offers the optimal solution to these algorithmic applications given an evaluation function. These individual set of devices usually individually referred as Processing Elements (*PEs*) or nodes make it possible to specialize processors to the target application. Such capacity is at the heart of recent gains in terms of system energy efficiency by exploiting applications properties (e.g. data or compute intensiveness) to occupy processing facilities at their maximum. Based on the differences between systems architectures, the level and form of heterogeneity can be formalized to describe a device and its processing capabilities. What complicates the creation of such a formalization is that while each *PE* has individual resources, they never work in complete isolation from one another since some sort of interconnection establishes communication between them. The main types of embedded processing elements available today are: *CPUs*, *GPUs*, *FPGAs* or *ASICs*. Contrary to *CPUs-GPUs* and *CPUs-FPGAs* couplings flourishing in products, the couple *GPU-FPGA* has not been much tested for its capacity to solve computational problems.

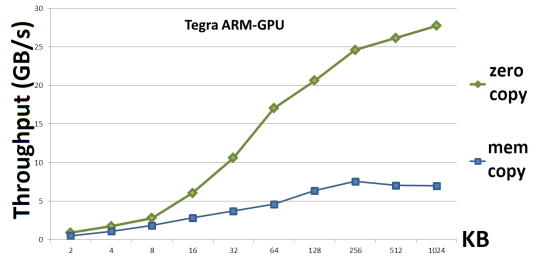
2. Related work

Heterogeneous computing has been a well researched domain for several decades where multiples non-von Neumann architectures started to aggregated and to exhibit great results specially on parallel programmable tasks [1]. However, like any novel hardware architecture proposal, heterogeneous platforms must overcome multiple challenges. One of the most known is the speed of interchip communication channels typically orders of magnitude lower than internal data accesses. Bittner et al. [2] and Thoma et al. [3] address this challenge by measuring throughput and latency as computation-to-communication ratio and establishing a direct communication between devices. In the area of embedded *GPU-FPGA* heterogeneous systems, Mohammad et al. [4] use a similar architecture with an Nvidia Jetson TX1 *SoM* with a Xilinx ZCU102 tested on multiple task such as histogram algorithm, dense and

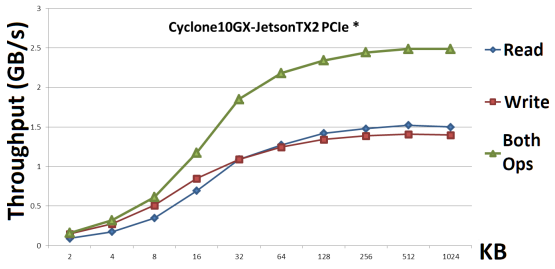
sparse matrix-vector multiplication, but not tested on deep learning tasks. Yuexuan et al. [5] also propose an architecture, but with an Nvidia Jetson TX2 *SoM* coupled with a Xilinx Nexys A7-100T, using a *CNN* for performance testing but with a *UART* interconnection interface, causing an overhead in the tensor transfer.

3. Early testing and results

Our proposed architecture consists of an Nvidia Jetson TX2 *SoM* as *CPU-GPU* system and a Intel Cyclone10GX *FPGA*. As the theoretical computation of each device is known, multiple sub-tasks or sub-data partitions must be empirically evaluated individually on each device. Multiple setups for communications must be also measured. The first approach consists in evaluating the communication between the devices, since multiple feature tensors have to be continuously transmitted. Figure 1 shows the communication throughput in GB/s per tensor size in KB of examples of *CNN* mappings on a *CPU-GPU* mapping and a *GPU-FPGA* mapping. As previously discussed, the device communication is orders magnitude slower than internal transfer on devices on the same die, i.e. *CPU-GPU*. Notice that techniques, such as, zero-copy can increase throughput, since data redundancy is avoided, taking advantage on shared memory.



(a) *CPU-GPU* throughput.



(b) *GPU-FPGA* throughput.

Figure 1: Communication throughput between devices.

We chose YOLOv2 [6] as an example of a large *CNN* workload where each layer is considered a subset of the total workload with an intrinsic set of resources and computation, usually defined as a partition. For this *CNN* model, as shown in Figure 2, the partitioning is not balanced since each layer has a exponentially reduced number of features when progressing along layers. This means that deeper layers could be easily handled for devices with memory constrains, but the gain in computation time should be hidden by communication time. A high computation-to-communication ratio is a necessary (but not sufficient) condition for having speedups on complex heterogeneous platforms. For increasing the computation-to-communication ratio, it is necessary to create large clusters of processing and reduce their data dependencies. Figure 3 shows a setup taking advantage of shallower layers on the *CPU-GPU* side, since the *GPU* memory is not as constrained as in the *FPGA*.

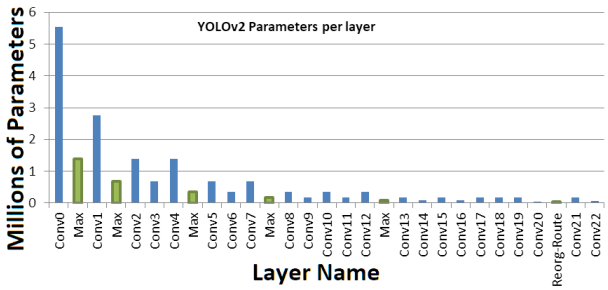


Figure 2: Layer-wise partition and number of parameters per layer.

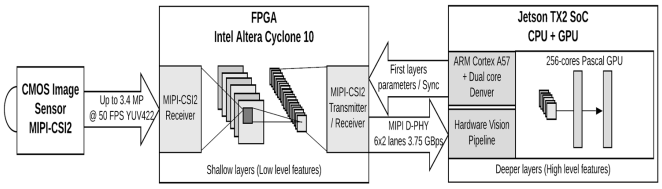


Figure 3: Proposed heterogeneous architecture with load partition.

4. Conclusions and future work

In this paper we have presented a *GPU-FPGA* architecture to be evaluated for performance and capabilities. It is built from two state-of-the-art device architectures, widely used in the *DL* community, and combines their advanced support of system parallelism. However, the most important challenge brought by such architecture is to efficiently partition the computation, represented with a model of computation, on the platform model of architecture. In future work, we will consider hardware-aware training and inference to consider cost parameters such as power consumption. The mathematical model of this optimization problem is still to be defined and therefore an optimization technique has to be selected to address this problem in a multi-constrained manner.

Acknowledgments

This project has received funding from the European Union's Horizon 2020 research and innovation programme under the Marie Skłodowska-Curie grant agreement No 765866

References

[1] M. Zahran, "Heterogeneous Computing: Here to Stay," *Communications of de ACM*, vol. 60, no. 3, pp. 42–45, March 2017.

[2] R. Bittner, E. Ruf, and A. Forin, "Direct GPU/FPGA communication Via PCI express," *Cluster Computing the Journal of Networks, Software Tools and Applications*, vol. 17, no. 2, pp. 339–348, June 2014.

[3] Y. Thoma, A. Dassatti, D. Molla, and E. Petraglio, "FPGA-GPU communicating through PCIe," *Microprocessors and Microsystems*, vol. 39, no. 7, pp. 565–575, October 2015.

[4] M. Hosseinabady, M. A. BinZainol, and J. Nunez-Yanez, "Heterogeneous FFPGA+GPU Embedded Systems: Challenges and Opportunities," *7th International Workshop on High Performance Energy Efficient Embedded Systems HIP3ES 2019*, April 2019. [Online]. Available: arXiv:1901.06331v2

[5] Y. Tu, S. Sadiq, Y. Tao, M. L. Shyu, and S. C. Chen, "A Power Efficient Neural Network Implementation on Heterogeneous FPGA and GPU Devices," July 2019.

[6] J. Redmon and A. Farhadi, "YOLO9000: Better, Faser, Stronger," *Computer Vision and Pattern Recognition (CVPR 2016)*, December 2016. [Online]. Available: arXiv:1612.08242v1