SqueezeJet-3: An Accelerator Utilizing FPGA MPSoCs for Edge CNN Applications

Panagiotis Mousouliotis, Ioannis Papaefstathiou, Loukas Petrou
School of Electrical and Computer Engineering, Aristotle University of Thessaloniki, 54124 Thessaloniki, Greece
Email: {pmousoul,ygp}@ece.auth.gr, loukas@eng.auth.gr

***Abstract*—Most FPGA-based Convolutional Neural Network (CNN) hardware accelerators target the datacenter rather than edge processing units. To further fill this gap, this work presents SqueezeJet-3 - a novel FPGA-based embedded system, consisting of software and hardware, for accelerating edge CNN inference. Even though SqueezeJet-3 is optimized for accelerating small ImageNet class CNNs, such as SqueezeNet v1.1, on low-end low-cost FPGA SoC devices, it can also be used for the acceleration of larger CNNs, such as the VGG16. Evaluation of our accelerator reveals better or comparable performance with that triggered by the current state-of-the-art similar tools and systems.**

SqueezeJet-3 is part of the CNN-Grinder framework [1], which is used to map CNNs to FPGA MPSoCs. Unlike other CNN-to-FPGA frameworks, which try to fully automate the conversion of a CNN to a description in C/C++ that can be handled by high-level-synthesis tools, CNN-Grinder exposes this procedure to the user. CNN-Grinder provides the user Matlab, C/C++, and HLS C/C++ templates to ease the mapping of CNNs to FPGA MPSoCs.

SqueezeJet-3 is a single computation engine architecture; it consists of a single computational block which is controlled by software and can accelerate CNN layers in a time-shared manner. SqueezeJet-3 supports the following CNN layer types: convolutions with kernel size 1×1 (padding 0), kernel size 3×3 (padding 1 or 0, stride 1 or 2), ReLU non-linearities, 3×3 max-pooling with stride 2, and 2×2 max-pooling with stride 2. Thus, the same design, which fits in low-end low-cost devices, such as the XC7Z020 and the ZU3EG by Xilinx, can be used to accelerate CNNs ranging from SqueezeNet v1.1 to VGG16. The accelerator's operation can also be easily adapted to support additional layers. Other features of the accelerator include: (1) use of double buffering to overlap computation (MAC operations) with communication (reading of new input activation pixel data), (2) dynamic fixed-point (DFP)[1] arithmetic support, (3) optional ReLU layer execution, and (4) high configurability; it is designed in a way that allows for seamless configuration of (a) the number of on-chip memories (OCMs) and processing elements (PEs), and (b) the representation accuracy (in bits) of the data.

The Xilinx Vivado HLS tool is used for the design of the accelerator and the Xilinx SDSoC tool is used for interfacing the accelerator to the processing system (PS) of the FPGA MPSoC. Figure 1 shows the interface between the PS and the accelerator, which resides at the programmable logic (PL) part of the FPGA MPSoC. High performance AXI ports, (i.e. the accelerator coherency ports (ACP) and the acceptance filter ID (AFI) port), are used for moving the activation and the weights-bias data respectively. General purpose (GP) AXI ports are used for moving the operations' scalar (SCA) values (kernel sizes, strides, etc.). Since some CNN layers are executed at the PS part of the FPGA MPSoC, as is the case with the concatenate layers of SqueezeNet v1.1, it is preferable for the activation data to be cache-coherent. Additionally, as it is shown in Figure 1, simple DMAs are used for moving the data from the PS memory system to the on-chip memories of the SqueezeJet-3 accelerator and vice versa. The block diagram of our accelerator is shown in Figure 2.

[1]http://lepsucd.com/ristretto-cnn-approximation/ristretto-approximation-schemes/

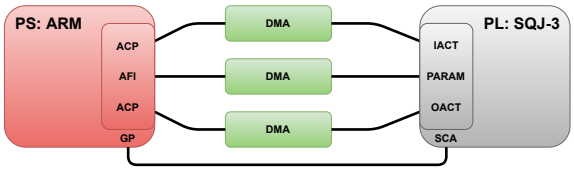

Fig. 1. Interface between SqueezeJet-3 and the PS.

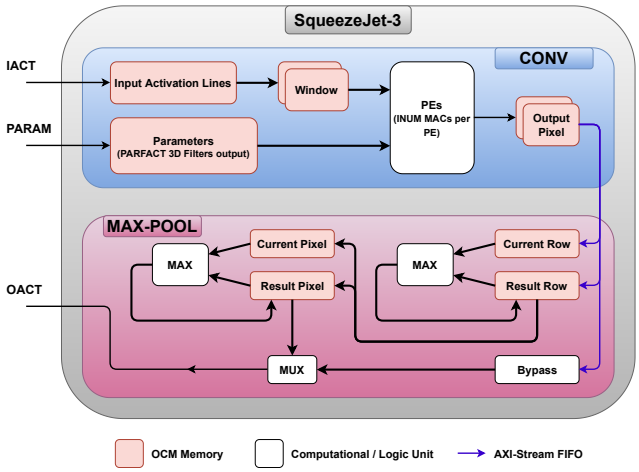

Fig. 2. Block diagram of the SqueezeJet-3 accelerator.

References

[1] P. G. Mousouliotis and L. P. Petrou, "CNN-Grinder: From Algorithmic to High-Level Synthesis Descriptions of CNNs for Low-end-low-cost FPGA SoCs," *Microprocessors and Microsystems*, 2020.

2576-2621/20/$31.00 ©2020 IEEE
DOI 10.1109/FCCM48280.2020.00069

IEEE computer society