# Data Movement Reduction for DNN Accelerators: Enabling Dynamic Quantization Through an eFPGA

Tim Hotfilter, Fabian Kreß, Fabian Kempf and Jürgen Becker
*Karlsruhe Institute of Technology*
Karlsruhe, Germany
hotfilter@kit.edu

Imen Baili
*Menta eFPGA S.A.S*
Valbone, France

***Abstract*—Computational requirements for deep neural networks (DNNs) have been on a rising trend for years. Moreover, network dataflows and topologies are becoming more sophisticated to address more challenging applications. DNN accelerators cannot adopt quickly to the constantly changing DNNs. In this paper, we describe our approach to make a static accelerator more versatile by adding an embedded FPGA (eFPGA). The eFPGA is tightly coupled to the on-chip network, which allows us to pass data through the eFPGA before and after it is processed by the DNN accelerator. Hence, the proposed solution is able to quickly address changing requirements. To show the benefits of this approach, we propose an eFPGA application that enables dynamic quantization of data. We can fit four number converters on an $1.5\,\mathrm{mm}^2$ eFPGA, which can process 400 M data elements per second. We will practically validate our work in the near future, with a SoC tapeout in the ongoing EPI project.**

***Index Terms*—heterogenous platforms, embedded FPGA, high performance computing, neural network accelerator**

Computational requirements of highly demanding applications, like the execution of deep neural networks (DNNs), have risen tremendously in recent years. DNNs are now able to solve tasks with increasing complexity, like semantic image segmentation [1] required for robotics or autonomous vehicles. To achieve this, state-of-the-art DNNs rapidly grow in parameter size and operations required. The latter surpassed 10 billion in 2020 [2]. As a result, execution of DNNs moved from CPUs and GPUs to dedicated hardware structures. Specialized hardware accelerators can leverage spatial parallelism in DNN operations, since most of them are independent. DNNs also offer potential for further optimization. For example, numerous research papers has demonstrated that quantization of parameters and intermediate results can reduce the hardware complexity [3] and thus save energy, while maintaining the network's accuracy. Furthermore, it has been shown that DNNs offer a high degree of sparsity. Dynamic pruning [4] of weights and feature maps can, hence, save much energy and latency.

While dedicated hardware accelerators can leverage ways to save a lot of energy and computations, they are usually not reconfigurable and expect a static dataflow. The need for adaptive accelerators becomes visible as machine learning engineers constantly present new models with more complex dataflows and structures to push the accuracy boundaries of all kinds of applications. However, hardware accelerators cannot adopt that quickly to the developments, since hardware design, testing and manufacturing is a time-consuming process.

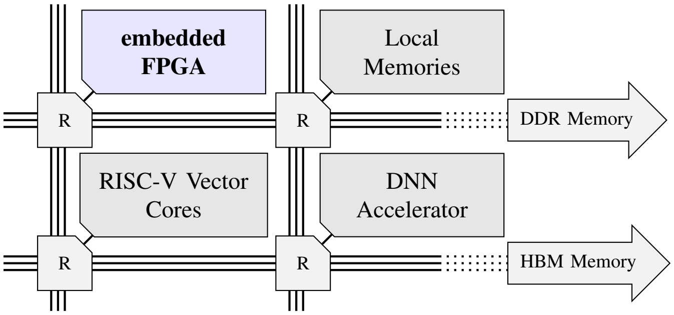

Fig. 1: Overview of the EPI accelerator. In the envisaged next generation, an embedded FPGA extends the accelerator with adaptable configurability to answer dynamic requirements.

In this paper, we therefore describe our approach to add flexibility to a powerful high performance computer (HPC) with an embedded FPGA (eFPGA) inside. The work is based on the HPC platform that is developed and investigated in the European Processor Initiative (EPI) [5]. The EPI platform features, besides general purpose CPUs, an EPI accelerator, which is designed in the second project phase stated in 2022 (Fig. 1). The EPI accelerator features multiple subcomponents: From a stencil accelerator for efficient and fast DNN execution to variable-precision RISC-V cores, which are also equipped with vector instructions. An eFPGA provided by Menta is added to the architecture to allow users to add applications that are not foreseen at chip build time or to react to dynamic changing requirements. All components are tightly linked with an on-chip network-on-chip (NoC) that also provides access to external DDR and HBM memories.

FPGAs to aid HPC computers have already been investigated in many paper, as well as DNN acceleration on FPGAs. For DNNs, Cichiwskyj et al. [6] first looked at runtime reconfiguration for DNN workloads, improving the typically high reconfiguration delay by splitting the accelerator into smaller components. Escobar et al. [7] looked at a broad range of more general HPC applications, pointing out the benefit but also the limitations of FPGAs in HPC. Research on eFPGAs was mostly carried out in embedded systems like microcontrollers and small ASICs [8] revealing their potential to add flexibility to the solutions. However, these eFPGAs are not coupled with high-performance glue-logic accelerators.

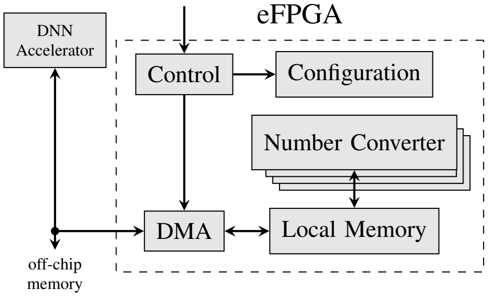

Fig. 2: Architecture overview that reshapes weights and input data before they are send to or read from the main memory to save valuable memory bandwidth.

DNN execution is highly dataflow driven and usually memory bounded. Quantization aims at reducing the bit width of both weights and feature map data, and can thus reduce also the bandwidth requirements for the memory interface. However, static quantization for all data is always a trade-off as some layers and regions in DNNs offer potential for further quantization, while others diminish the network accuracy with higher quantization. Thus, dynamic quantization has shown tremendous benefits. Song et al. [9] present, for instance, a tailored accelerator for dynamic quantization that saves 72% energy with less than 1% accuracy loss. However, they had to design an architecture for this particular application.

Since dynamic quantization offers great benefit to make DNNs more energy efficient and their memory footprint smaller, we will explore how we can utilize the eFPGA to enable dynamic quantization as an exemplary application to demonstrate the benefits of the eFPGA. In this case, the eFPGA can work together with an unmodified DNN accelerator. Therefore, we assume a DNN accelerator that operates on the well-established bfloat16 format, a 16-bit floating point format optimized for DNNs. To save energy-intensive memory transfers, we use the eFPGA to reshape data when it is fetched from and stored to the off-chip memory. Therefore, we propose the hardware architecture depicted in Fig. 2. Before a layer gets processed, we first load a configuration to our architecture, featuring the number conversion format, e.g. convert from 6 bit integer to bfloat16. A previously configured DMA then fetches data from the off-chip memory and puts it in a local scratchpad memory. Configurable number converters perform a transformation from an arbitrary integer bit-width to bfloat16 on the values in the local memory. Once the conversion is done, we send the results to the local memory of the DNN accelerator via the DMA and the on-chip network. The same applies for the results generated by the accelerator the other way round. We do this in a highly parallel fashion, to maintain high troughput. Although our approach adds latency to the overall system, we can save numerous bytes of data movement. In addition, this approach enables us to dynamically adapt the bit width, facilitating a high degree of quantization while keeping a high accuracy as well.

EPI phase 2 stated in 2022 and is currently ongoing. Our preliminary results show that we can map four number converters, the control logic and the DMA on an eFPGA with $1.5\,\mathrm{mm}^2$ total area using a 12 nm TSMC process. Therefore, we configured the eFPGA resources to fit the application before hardening, as described in [10]. With our application running at 100 MHz, we can achieve a theoretical trough-put of 400 M word/s, which is sufficient for fast DNN acceleration. Assuming 8-bit memory numbers, we can cut the bandwidth requirements for the external memory in half.

DNNs offer great potential for optimizing the high number of computations and the memory transactions. Our work addresses these challenges in a highly adaptable way by exemplary means of a number converter to enable dynamic quantization. To validate our architecture design, we will implement it on a test setup, to adjust the bandwidth requirements for the eFPGA and the accelerator and to validate the theoretical through-put figures. In the future, it is also conceivable to cover applications like pruning and leveraging sparsity in feature maps to increase the performance further. The EPI consortium will tapeout the chip, hence, we will be able to demonstrate the full potential of our approach on real hardware in the near future.

## Acknowledgements

This project has received funding from the European High Performance Computing Joint Undertaking (JU) under Framework Partnership Agreement No 800928 and Specific Grant Agreement No 101036168 (EPI SGA2). The JU receives support from the European Union's Horizon 2020 research and innovation programme and from Croatia, France, Germany, Greece, Italy, Netherlands, Portugal, Spain, Sweden, and Switzerland.

## References

[1] Y. Huang, Q. Wang, W. Jia, and X. He, "See more than once – kernel-sharing atrous convolution for semantic segmentation," no. arXiv:1908.09443, Nov 2019, arXiv:1908.09443 [cs].

[2] W. Liu *et al.*, "Fastbert: a self-distilling bert with adaptive inference time," no. arXiv:2004.02178, Apr 2020, arXiv:2004.02178 [cs].

[3] A. Gholami *et al.*, *A Survey of Quantization Methods for Efficient Neural Network Inference*, Jun 2021, no. arXiv:2103.13630.

[4] A. Ankit, T. Ibrayev, A. Sengupta, and K. Roy, "Trannsformer: Clustered pruning on crossbar-based architectures for energy-efficient neural networks," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 39, no. 10, p. 2361–2374, Oct 2020.

[5] M. Kovač, P. Notton, D. Hofman, and J. Knezović, "How europe is preparing its core solution for exascale machines and a global, sovereign, advanced computing platform," *Mathematical and Computational Applications*, vol. 25, no. 33, p. 46, Sep 2020.

[6] C. Cichiwskyj, C. Qian, and G. Schiele, "Time to learn: Temporal accelerators as an embedded deep neural network platform," in *IoT Streams for Data-Driven Predictive Maintenance and IoT, Edge, and Mobile for Embedded Machine Learning*. Springer International Publishing, 2020, p. 256–267.

[7] F. A. Escobar, X. Chang, and C. Valderrama, "Suitability analysis of fpgas for heterogeneous platforms in hpc," *IEEE Transactions on Parallel and Distributed Systems*, vol. 27, no. 2, p. 600–612, Feb 2016.

[8] F. Renzini, C. Mucci, D. Rossi, E. F. Scarselli, and R. Canegallo, "A fully programmable efpga-augmented soc for smart power applications," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 67, no. 2, p. 489–501, Feb 2020.

[9] Z. Song, B. Fu, F. Wu, Z. Jiang, L. Jiang, N. Jing, and X. Liang, "Drq: Dynamic region-based quantization for deep neural network acceleration," in *2020 ACM/IEEE 47th Annual International Symposium on Computer Architecture (ISCA)*, May 2020, p. 1010–1021.

[10] T. Hotfilter *et al.*, "Towards reconfigurable accelerators in hpc: Designing a multipurpose efpga tile for heterogeneous socs," in *2022 Design, Automation Test in Europe Conference Exhibition (DATE)*, p. 628–631.