



The Omnitek DPU

An FPGA-based Deep-Learning Processing Unit that achieves World-Class Performance per Watt

Architecture, Tool Flow and Benchmark Figures

Roger Fawcett, CEO Omnitek

Neural networks have been shown to offer a powerful architecture for addressing many types of machine learning problems. Deep Neural networks (DNNs) provide an adaptable mapping function from a large set of input training data to a set of desired results.

DNN computation is not well-suited to the Von Neumann computer architecture at the heart of CPUs. It is better suited to architectures with distributed, massively-parallel computation and local memory. So, the race is on to develop highly-effective processing units on which to run neural networks.

Whether designing for data centres or embedded applications, the end users will have a wish-list of attributes, including:

- *High throughput*
- *High performance per watt*
- *Low latency*
- *Accurate inferences*
- *Low component costs*
- *Commercially viable development times and low development costs*
- *Adaptability, both to different tasks and to advances in machine learning technology*

The first alternative technology to find favour was the GPU, thanks to the high level of parallelism needed both for their primary role of processing graphics and to process neural networks. Other developers favoured the use of dedicated ASICs. However different DNN topologies (such as CNNs and RNNs) vary in their compute requirements which results in different architectures for the optimal compute engines. In addition, different data types (such as reduced precision integer and floating point) have proven effective for different DNNs.

This area is under constant change as research unveils new approaches. The overall architecture of DNNs and machine learning techniques seem set to evolve for many years to come. As a consequence, it is important to focus on technologies that can adapt to varying workloads and adopt new techniques as they emerge from research. Fixed ASIC or GPU architectures are unlikely to be able to adapt to future requirements, which is why we have chosen to use FPGA as the only architecture with the ability to completely rewire.

In this paper we present benchmark figures for the Omnitek DPU (Deep-Learning Processing Unit) showing its leading performance in comparison with existing ASICs and GPUs. We also set out a philosophy for Artificial Intelligence (AI) acceleration based around FPGAs, combined with research which we believe will be the optimum strategy going forward. We also show that the supporting Omnitek DPU SDK (Software Development Kit) provides the ability for users to program the DPU using traditional C/C++ or Python using standard frameworks such as TensorFlow, without the need for any FPGA knowledge and with compile times of only a few seconds.

Omnitek DPU: FPGA-based Deep-Learning Processing Unit

Contents

| | | |
|-----|--|----|
| 1 | Introducing the Omnitek DPU..... | 3 |
| 1.1 | The Vision behind the Omnitek DPU..... | 3 |
| 1.2 | General Advantages of FPGA..... | 4 |
| 2 | The Omnitek DPU Architecture..... | 5 |
| 2.1 | CNN Architecture..... | 5 |
| 2.2 | Other DNN Topologies..... | 6 |
| 2.3 | Resource Efficiency..... | 7 |
| 2.4 | Modular Architecture..... | 7 |
| 3 | DPU Software Development Kit and Tool Flow..... | 9 |
| 3.1 | Use of Familiar Neural Network Tools..... | 9 |
| 3.2 | Microcode Overlay..... | 11 |
| 3.3 | The Resulting Flow..... | 11 |
| 4 | Benchmark Performance Figures..... | 13 |
| 5 | Other Perspectives..... | 14 |
| 5.1 | Accuracy..... | 14 |
| 5.2 | DNN Compute Efficiency..... | 14 |
| 5.3 | Adaptability to different applications..... | 15 |
| 5.4 | Business Considerations: Part Cost and Time to Market..... | 15 |
| 5.5 | Adopting Future Silicon Technology Nodes..... | 15 |
| 6 | Research Program with Oxford University..... | 16 |
| 7 | Conclusions..... | 16 |
| | Related Papers..... | 16 |

1 Introducing the Omnitek DPU

Omnitek's new DPU Suite provides the components needed to implement inference engines for Deep Neural Networks (DNNs) on FPGAs such as the Intel Agilex, Stratix 10 and Arria 10 FPGAs. The key features are:

- Ability to define the required neural network topology in C/C++/Python using primitives from a neural network framework such as TensorFlow in exactly the same way that you might define a network topology for implementation on a GPU, for example.
 - No FPGA knowledge is required
 - No FPGA synthesis or 'Place & Route' is required
 - Instead, our compilation and quantization tools generate microcode which is compiled from the C/C++/Python software in seconds
- Highly flexible design:
 - Architecture optimised for the application workload (eg. CNN, RNN/LSTM/MLP or variants) and programmed for each task through the above microcode
 - Scalable design with each DPU comprising a selectable number of separate DPU engines, allow the user to optimise the trade-off between performance and power/cost
 - Additional functions e.g. video/vision processing IP can readily be added alongside the DPU, sharing resources such as on-board memory and PCIe DMA as required
 - Novel topologies can readily be incorporated as they emerge from industry and academia, as a consequence of using FPGA technology
- Use of FPGA resources optimised to deliver the best possible performance out of the selected FPGA at the lowest cost and the lowest power consumption per inference
- Numerical precision chosen to optimise performance with no loss of accuracy

Furthermore, the Omnitek DPU is equally able either to be implemented in an FPGA with an embedded processor such as is provided by members of the Intel Agilex, Stratix 10 and Arria 10 SoCs or to be used in a PC or an environment such as a Data Centre where multiple copies of the DPU are implemented in a large FPGA on a PCIe card.

1.1 The Vision behind the Omnitek DPU

A major issue underlying the use of neural networks is that they are not well suited to the Von Neumann architecture of traditional CPUs; they run slowly and achieve poor performance per watt when users of these networks really require:

- Low, deterministic latency together with the highest possible throughput
- The best possible performance per watt
- The ability to integrate the network with other processing functions
- Ability to produce optimum performance across a range of different current and as yet unknown future machine learning workloads

CPU performance has been significantly improved due to native support for AI operations, such as the DL Boost. However, FPGAs can still provide significantly better performance. Omnitek's vision for its DPU is as a processing unit for machine learning that:

- Can be used to implement all types of neural network topologies – convolutional neural networks (CNNs), recurrent neural networks (RNNs and LSTMs), and multi-layer perceptrons (MLPs)
- Delivers world-class performance, both in terms of inferences per second and low power consumption, across different neural networks, through being optimised for both the architecture and the workload
- Offers a software-centric interface so that it can be set up to deliver the required results by a user with no knowledge of firmware implementation on either FPGAs or ASICs and whose knowledge of working with different neural networks is based on tools such as TensorFlow, OpenVINO and Caffe

Omnitek DPU: FPGA-based Deep-Learning Processing Unit

- Can be integrated with other IP Cores (such as video connectivity and processing) to create complete SoC DNN solutions
- Is tied into the latest research and will be able to adapt to implement future compute requirements

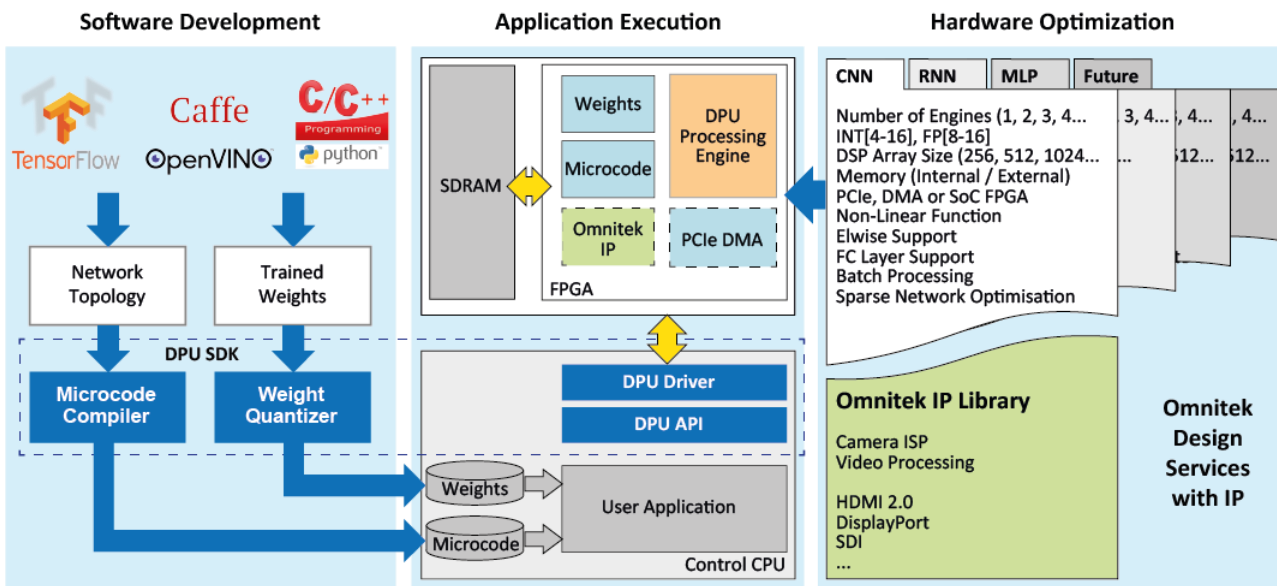


Figure 1: Simplified representation of the DPU concept, presenting the process of software development on one side with the various options for configuring the DPU hardware on the other

1.2 General Advantages of FPGA

Omnitek sees FPGAs as the best technology in which to implement the compute engine for neural networks for reasons including:

- Features such as large numbers of DSP slices and distributed memory storage, plus significant programmable logic
- The ease with which the FPGA can be re-programmed, which in turn allows it to be optimised to different tasks and workloads
- Its ability to handle values expressed with arbitrary precision
- The ease with which firmware offering additional functions can be integrated alongside the principal components; for example, for a vision application it may be required to add video I/O connectivity and warp processing
- The range of FPGA sizes and features means that IP can rapidly be targeted to low power embedded applications or large data centre processing devices
- The ability to transfer FPGA IP to the latest technology node.

Forthcoming FPGA architectures also promise more logic running at higher clock speeds and features such as High-Bandwidth Memory (HBM), which is expected to deliver around 10x the bandwidth achieved with earlier memory interfaces, and software-programmable elements that are predicted to deliver up to 20x better performance in machine learning tasks.

2 The Omnitek DPU Architecture

2.1 CNN Architecture

The following diagram shows the key processing elements of the Omnitek DPU when implementing a convolutional neural network.

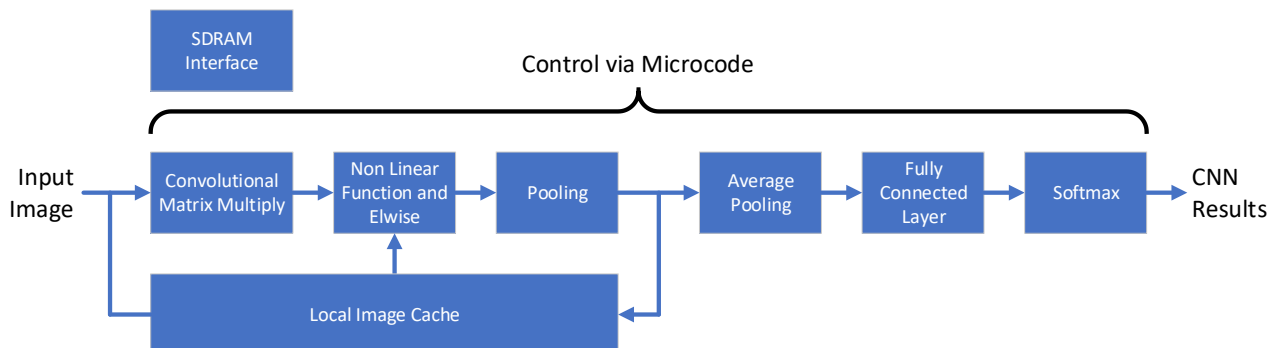


Figure 2: Simplified representation of the architecture used for Convolutional Neural Networks

This generic form has the following key properties:

- It is capable of being programmed via microcode to implement a wide range of CNN topologies such as GoogLeNet, ResNet, VGG, MobileNet etc.
- It can run all of these CNNs with high efficiency
- It employs a unique combination of low-precision fixed point maths and floating-point maths to achieve a very high compute density without loss of accuracy.
- In the Intel Arria 10 GX 1150, it uses 93% of the available DSPs while using 61% of the general logic – and the DSPs can be clocked at 87% of the maximum theoretical clock frequency.
- All other functionality can be implemented in parallel with the main convolutional matrix multiply, thus keeping this block busy all the time
- It manages external I/O of images via PCIe and other data via SDRAM such that the performance is never limited by the bandwidth of these external interfaces

This architecture is optimised for implementation on Intel Arria 10 and Stratix 10 and Agilex FPGAs and SoCs.

Specific implementations will employ one or more DPU engines running in parallel in a single FPGA. This is well illustrated by an example implementation in an Intel Arria 10 GX 1150.

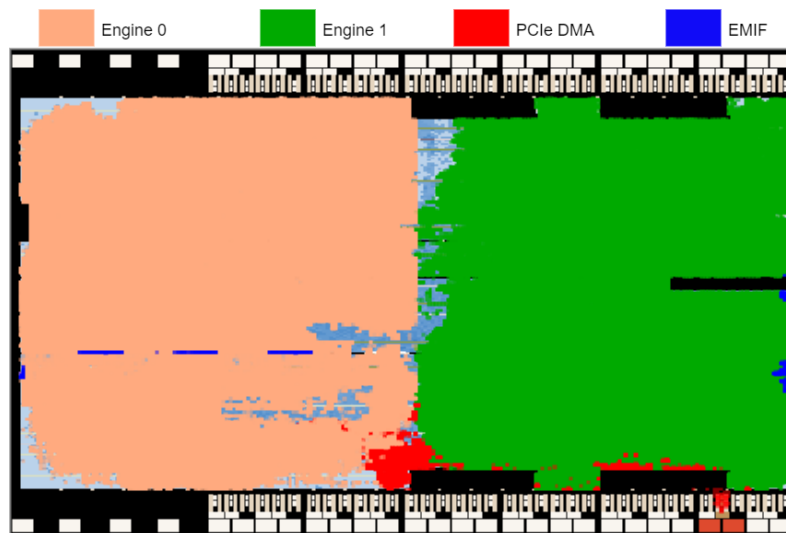


Figure 3: 2-Engine Build on GX 1150

2.2 Other DNN Topologies

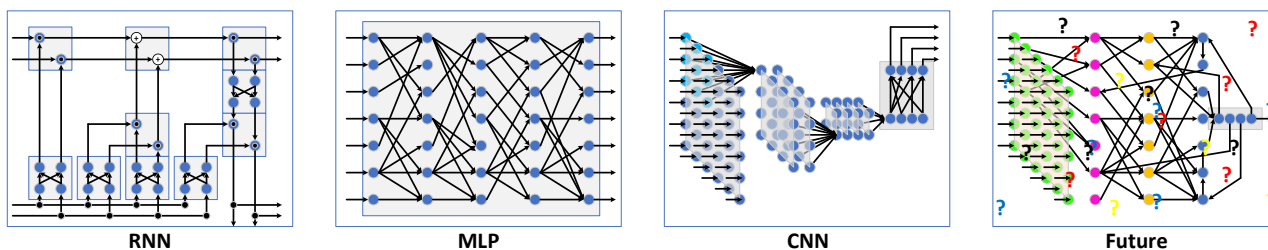


Figure 4: Current and future DNN topologies

The diagrams in Figure 4 show simplified schematics of common DNN topologies, including:

- RNNs (Recurrent Neural Networks), which includes LSTMs (Long Short Term Memories: *shown*)
- MLPs (Multi-Layer Perceptrons)
- CNNs (Convolutional Neural Networks)
- Unknown future topologies (*imagined*)

The initial release of the DPU provides a generic compute platform for CNNs. As such, many of the techniques and processing blocks are only applicable to CNNs. For example, CNNs have the following properties which can be exploited:

- Ability to operate with minimal or no loss of accuracy using more efficient mathematical frameworks employing reduced precision
- Shared weights in the convolutional matrix multiply
- Max Pool and Average Pool – features which are generally not used outside of CNNs
- Varying sized fully connected layers: the weights used in these layers typically contain significant redundancy and can be pruned by up to 90% without any significant loss of accuracy
- Use of relatively large feature caches for intermediate results
- Use of the ReLU function means that intermediate results are generally unsigned
- EIWise Add between intermediate feature maps (first introduced in the original ResNet paper)

Most of these features do not apply to RNNs and MLPs. However, these architectures have other unique features including:

- Precision requirements for optimum results that differ from those of CNNs

Omnitek DPU: FPGA-based Deep-Learning Processing Unit

- Extensive use of ElWise Add and Multiply functions
- Intermediate feature maps that differ in size to those of CNNs
- Large, Fully-Connected layers that encompass most of the compute demand
- Significant weight redundancy
- Unique weights per multiply
- External memory bandwidth requirements that differ from those of CNNs
- Recurrent feedback loops

Omnitek has an extensive research program in place which is yielding roadmap architectures that optimally exploit these features to achieve maximum compute performance. FPGAs are the only compute platform that can be dynamically rewired to exploit these different features at a fine-grained logic level.

DNNs and more general machine learning architectures have undergone vast change in the past 10 years due to a massive research effort in both industry and academia. As depicted on the right of Figure 4, it is difficult to predict what future requirements will be. However, past experience indicates that FPGA-based solutions will be able to adapt the most rapidly and the most efficiently to these requirements.

2.3 Resource Efficiency

The DPU design employs a novel mathematical framework which is highly compatible with the architecture of the Arria 10, Stratix 10 and Agilex architectures, leading to excellent resource efficiency and great compute density. The low-precision fixed-integer multipliers may be implemented individually in soft logic or combined with other multipliers in the DSP blocks, while the floating-point operations are performed in Intel's unique IEEE 754 compliant floating-point DSP blocks.

The following table gives usage figures for 2 DPU engines in a GX 1150.

Note the use of 93.3% of the on-chip DSPs, a key element in the computational operations needed by any neural network, and at the same time, the relatively low use of ALMs and Flip-Flop elements alongside supporting elements such as EMIFs and a PCIe DMA controller. This low logic utilization eases place and route of the design and enables Omnitek to add additional functionality in the future.

Table 1: Resource usage of a 2-Engine DPU implementation in an Arria 10 GX 1150

| | GX 1150 Resources | 2 DPU Engines | | EMIF and PCIe DMA | | DPU Totals | |
|------|-------------------|---------------|-------|-------------------|------|------------|-------|
| DSP | 1,518 | 1,416 | 93.3% | 0 | 0% | 1,416 | 93.3% |
| ALM | 427,200 | 249,018 | 58.2% | 5,210 | 1.2% | 260,953 | 61.1% |
| FF | 1,708,800 | 756,102 | 43.6% | 10,974 | 0.6% | 776,426 | 45.4% |
| M20k | 2,713 | 1,864 | 64.1% | 91 | 3.4% | 1,955 | 64.1% |

2.4 Modular Architecture

The CNN version of the DPU comprises multiple DPU engines. These engines are configurable in size, with the initial version comprising 4,096 MACs (constructed using DSPs or logic) and associated logic and RAM elements. These DPU engines have been designed to be floor-planned into physically separate areas of an Arria 10 GX, Stratix 10 GX or Agilex FPGAs.

Making the DPU fit in a compact physical region doesn't just keep the paths within the DPU suitably short (and hence suitably speedy), it also means that a highly usable region can be made available for other FPGA IP simply by leaving one DPU out of the overall design. This is key where the DPU needs to be implemented alongside other IP to deliver more comprehensive data centre or edge compute applications.

Omnitek DPU: FPGA-based Deep-Learning Processing Unit

The resulting layout is represented in the following diagram which illustrates a GX 1150 populated with 1 DPU engine, with the other half of the device available for other FPGA IP.

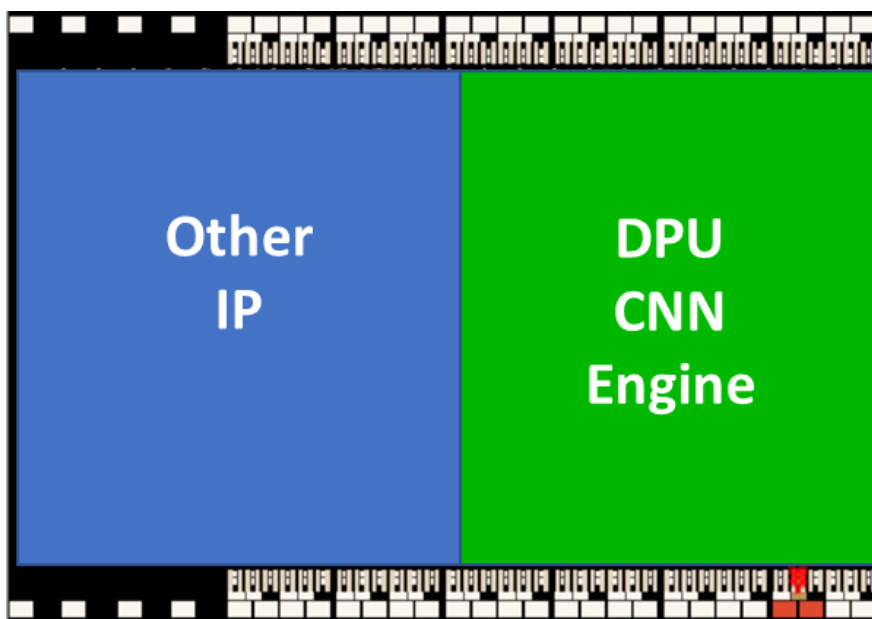


Figure 5: Representation of example GX 1150 build with space in one half for other FPGA IP

The performance of the remaining DPU engine won't be affected because this additional IP is kept within a defined area of the FPGA.

Another useful feature of the method Omnitek has adopted in implementing its DPU is that data is shared between the different DPU engines implemented in the FPGA. This data is ultimately fed in and out of the chip either through the FPGA's hard PCIe IP or to and from SDRAM through an on-chip EMIF (External Memory Interface), but it is distributed around the DPUs through a 'Network on Chip' ('NoC') as illustrated in the following diagram.

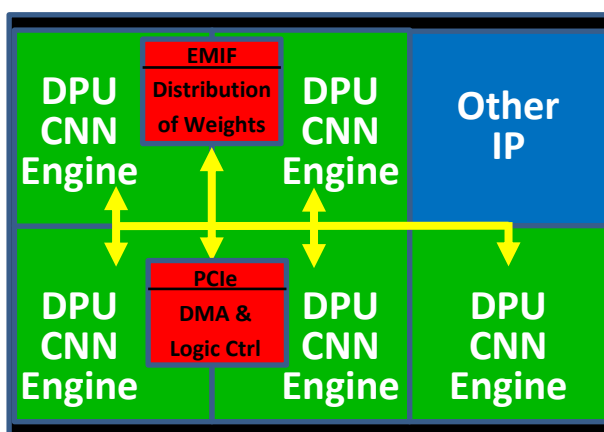


Figure 6: Representation of a 5-Engine Stratix 10 GX build showing EMIF, PCIe and NoC

The low logic utilization of the DPU engines mean that they can easily be placed over these infrastructure components while still achieving timing closure at the peak clock frequency. This is illustrated by the P&R floorplan shown in Figure 3.

The design also uses the Omnitek PCIe Streaming DMA Controller which offers independent flow-controlled memory-based or FIFO-based DMA and achieves all the necessary host communications in an extremely small footprint. Similarly, the NoC uses very little FPGA resource, again allowing the DPU engines to be placed and routed over the top of these infrastructure components.

Any additional FPGA IP that the user chooses to add can either make use of the NoC to share the EMIF and PCIe resources used by the DPUs (along with supporting elements such as Omnitek's PCIe DMA controller) or make its own arrangements.

3 DPU Software Development Kit and Tool Flow

3.1 Use of Familiar Neural Network Tools

In contrast to the hand-crafted RTL used to implement the individual DPU engines, Omnitek has taken a software-centric approach to the user's task of producing the model of the required neural network topology and determined that all tools that need to be used will be the ones familiar to any user with experience of modelling neural networks, for example for implementation on a CPU or GPU.

Neural network models for use with the DPU are therefore:

- Written in C, C++ or Python
- Using neural network constructs offered by standard frameworks such as TensorFlow

Indeed, a neural network model prepared for use on the Omnitek DPU using TensorFlow can both be used with all the facilities such as TensorBoard offered within TensorFlow and be used with any of the inference engines supported by TensorFlow.

Omnitek DPU: FPGA-based Deep-Learning Processing Unit

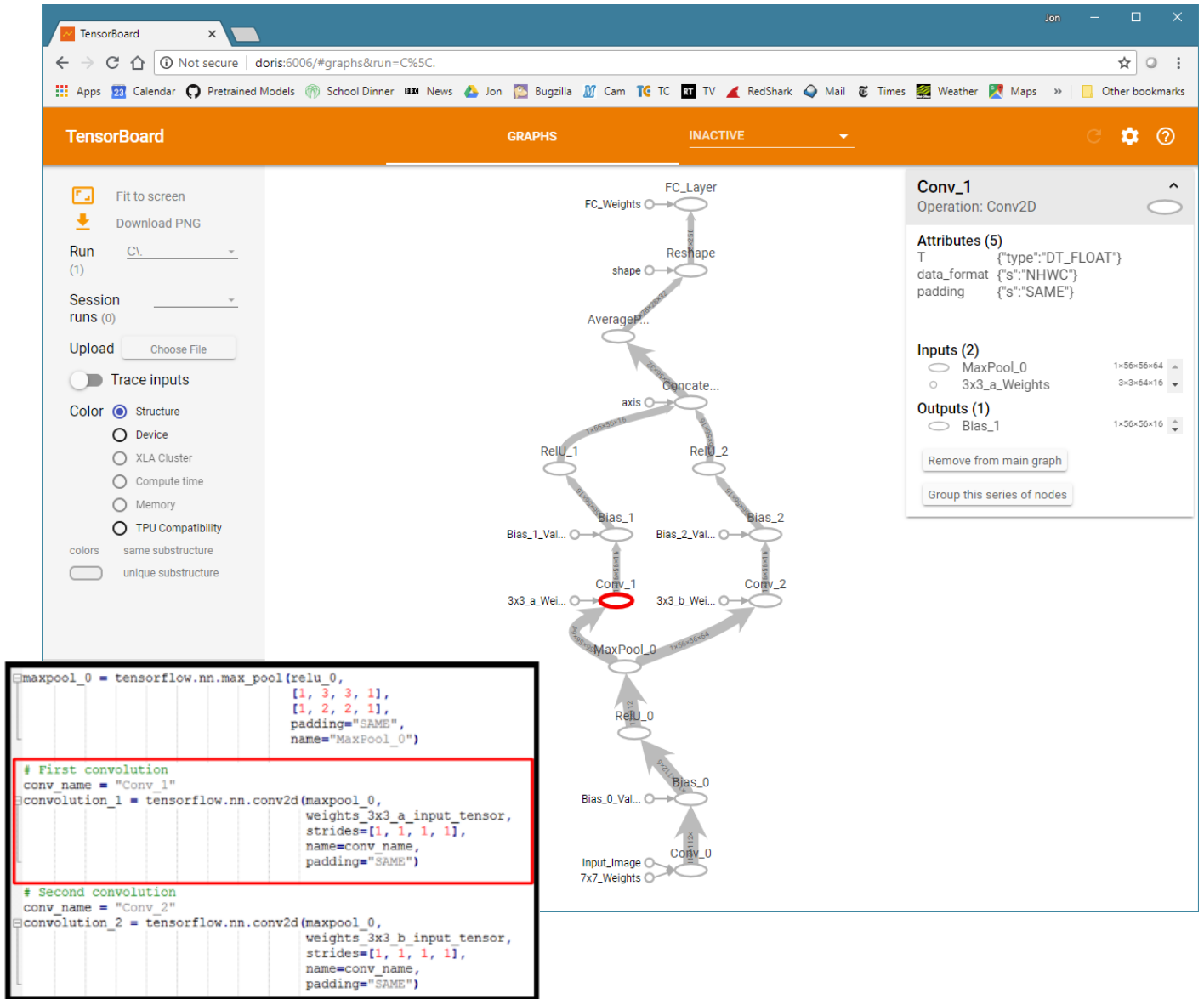


Figure 7: TensorBoard representation of one of Omnitek's demo networks. The box on the left is a snippet of the script, showing the definition of the Conv_1 node outlined in red.

3.2 Microcode Overlay

Each DPU engine is essentially a block of firmware delivering an FPGA subsystem such as that illustrated in Figure 2 (on page 5). Each convolutional operation can be optionally combined with additional parallel processing such as ReLU, pooling and ElWise residual add.

Each of these computation blocks can be programmed. For example, the convolution can operate on an arbitrary feature map with arbitrary weight tensor sizes and applying an arbitrary stride. Other components can be similarly parameterized.

The way this is achieved is through the use of microcode that defines the required configuration for each cycle through the FPGA subsystem. This microcode is downloaded to the FPGA by the application which runs the task in much the same way as a computer program is downloaded to a CPU. As each cycle completes, the microcode for the next cycle is read and the firmware configured in response.

The microcode can be regarded as overlaying the installed firmware and it is generated by compiling the C/C++/Python software, that defines the required neural network topology using the DPU compiler, included in the Omnitek DPU SDK (delivered as part of the DPU Suite). Compilation takes a few seconds, compared with the very much longer synthesis times traditionally experienced with FPGA design, and downloading the microcode to the FPGA ready for use is just as quick.

Our use of microcode has a major advantage over traditional FPGA design in that it means no FPGA knowledge is required, but it is not the only approach to offer that advantage. An alternative route is provided by high-level synthesis languages such as the high level synthesis (HLS) of C-based languages directly into RTL (i.e. VHDL or Verilog). This is an alternative way to open up FPGAs to software programmers and enables rapid prototyping and verification.

While the use of HLS is a laudable goal with many advantages, Omnitek believes that significantly more optimised FPGA implementations can be produced by FPGA experts working with traditional RTL. Omnitek's microcode approach means that users of the Omnitek DPU get to work with FPGA implementations hand-crafted by Omnitek that still can be programmed to deliver the required neural network in a few seconds, starting from the regular C/C++/Python + TensorFlow software defining the required neural network topology.

3.3 The Resulting Flow

The approach Omnitek has taken means the route for the user from selecting a neural network topology to using this neural network to generate inferences reduces to simply:

- Obtain or create a model of the network topology in C/C++/Python;
- Obtain or create a set of weights for the type of analysis they wish to make;
- Issue a couple of command line instructions – one to generate the required microcode from the model, the other to 'quantize' the weights for execution by the DPU; and
- Run an 'Inference App' (also written in C/C++/Python) that loads the microcode into the DPU, loads the weights, reads in the items to be analysed and presents the inferences made

The diagram overleaf takes a more detailed look at the flow, from deciding on the neural network to use through to using the DPU to make inferences. It shows the steps taken when TensorFlow is used as the creation framework. The steps needed will be similar where other creation frameworks are used, such as Caffe.

Omnitek DPU: FPGA-based Deep-Learning Processing Unit

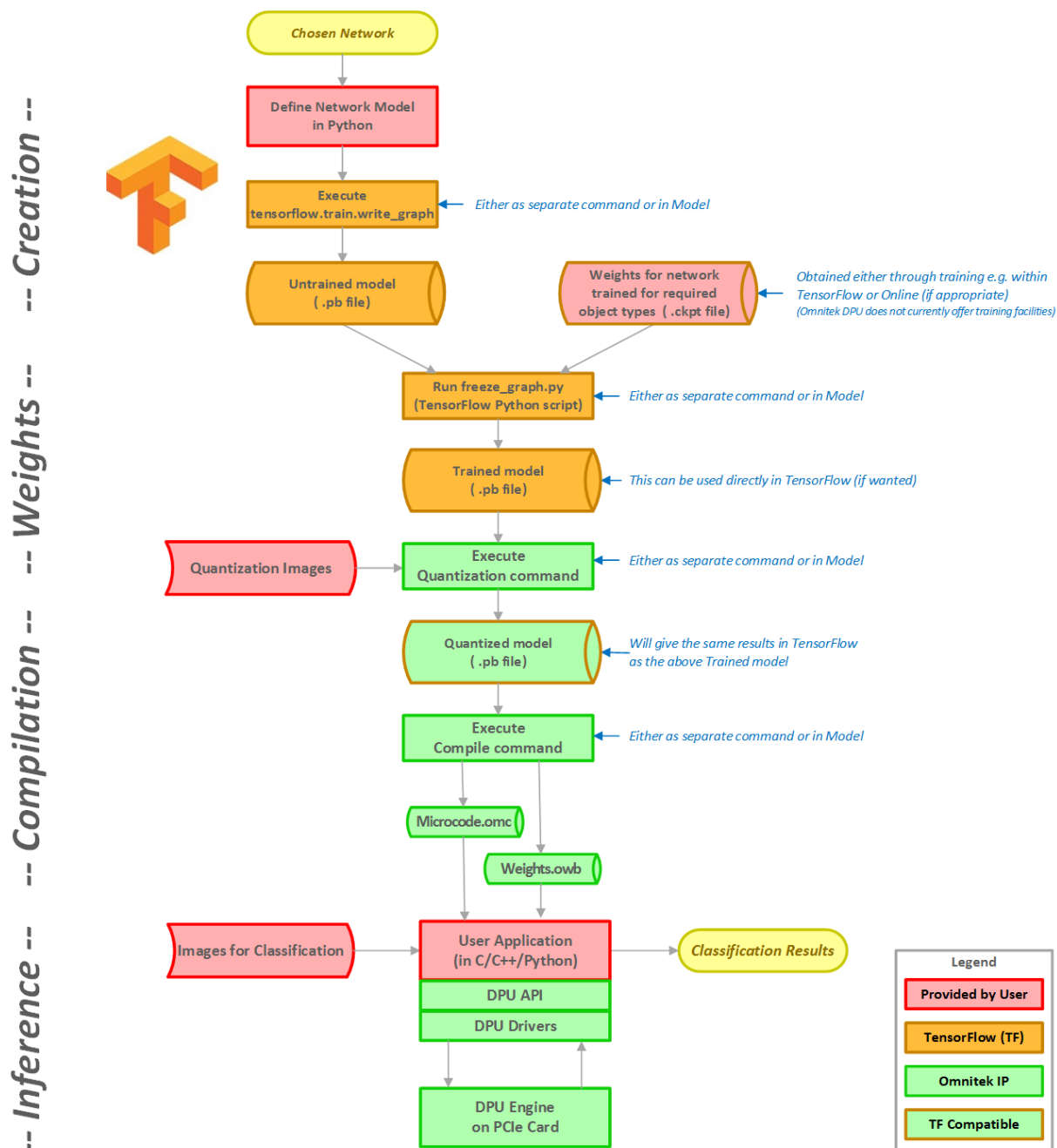


Figure 8: Outline Flow when TensorFlow is used.

Optimal operation will also require reprogramming of the underlying firmware when changing between using a CNN-type network and an RNN-type network (for example), but Omnitek makes this something that happens quickly and efficiently in the background, and without intervention from the user through having multiple FPGA bitstreams available to be downloaded into the FPGA automatically, each implementing the configuration needed for a particular type of network.

4 Benchmark Performance Figures

The following table shows a summary of the current benchmark figures for three popular CNNs: GoogLeNet, VGG16 and ResNet 50, when running on an Arria 10 Gx1150, with a single bank of DDR4 SDRAM.

| | ips | TOPs | GOPs/W | Efficiency |
|------------------|----------|------|----------|------------|
| Peak | | 6.55 | 145.6356 | 100% |
| GoogLeNet | 1535.892 | 4.85 | 107.8169 | 87% |
| VGG16 | 198.0619 | 6.08 | 135.0926 | 95% |
| ResNet 50 | 662.0352 | 5.11 | 113.6052 | 83% |

Throughput is often quoted as a number of inferences per second (IPS). However, comparisons based on IPS values are plainly only valid where all the inferences being considered are running the same neural network, so more general comparisons typically look at the numbers of Tera-operations per second (or 'TOPs') achieved.

The achieved TOPs figure for a particular network can be compared to the theoretical maximum. As can be seen, the DPU achieves up to 95% of this efficiency, depending upon the network.

Figures for Stratix 10 and Agilex devices will significantly exceed these figures. What is impressive about the Arria10 figures is that an exceptionally high performance has been achieved for a comparatively low cost FPGA device.

The energy efficiency of compute engines is conventionally expressed in terms of performance per watt,. In the case of inference engines, billions of operations are carried out per second and so the energy efficiency is quoted in giga-operations per second per watt ('GOPS/W').

The bigger the number of GOPS/W achieved the better, as it means the power needed to process the same number of neural network operations will be less.

5 Other Perspectives

5.1 Accuracy

Omnitek has designed a novel mathematical framework which employs reduced precision without compromising accuracy. The reduced precision results in much lower resource usage and hence a much higher compute complexity and performance per watt.

Specifically, the first layers of the CNN operate at 8-bit precision while subsequent layers operate at lower precision. All layers take advantage of additional floating-point processing which significantly increases the accuracy. The resulting overall accuracy for the CNN is within one percentage point of the original FP32 accuracy without retraining.

To recover the lost accuracy Omnitek has developed a novel retraining technique which results in an accuracy equivalent to that of the original FP32 model. Furthermore, this retraining does not require access to the original labelled data and can actually be performed using unlabelled new data, making the retraining very convenient.

5.2 DNN Compute Efficiency

The excellent throughput and performance per watt figures of the Omnitek DPU are due to several factors:

- Ability to exploit over 90% of the DSPs in the GX 1150 device
- Ability to clock the DSPs at near maximum frequency
- An operating efficiency for the DSPs of between 83% and 95% (depending on choice of network model)

The operating efficiency is a measure of the number of useful multiply-accumulate cycles taken by the DSP blocks compared with the maximum peak performance.

Achieving a high efficiency across a wide range of tensor sizes and convolution operations is a challenge for many hardware architectures. Consider the following table from the Google TPU paper:

| <i>Application</i> | <i>MLP0</i> | <i>MLP1</i> | <i>LSTM0</i> | <i>LSTM1</i> | <i>CNN0</i> | <i>CNN1</i> | <i>Mean</i> | <i>Row</i> |
|------------------------------------|-------------|-------------|--------------|--------------|-------------|-------------|-------------|------------|
| Array active cycles | 12.7% | 10.6% | 8.2% | 10.5% | 78.2% | 46.2% | 28% | 1 |
| Useful MACs in 64K matrix (% peak) | 12.5% | 9.4% | 8.2% | 6.3% | 78.2% | 22.5% | 23% | 2 |
| Unused MACs | 0.3% | 1.2% | 0.0% | 4.2% | 0.0% | 23.7% | 5% | 3 |
| Weight stall cycles | 53.9% | 44.2% | 58.1% | 62.1% | 0.0% | 28.1% | 43% | 4 |
| Weight shift cycles | 15.9% | 13.4% | 15.8% | 17.1% | 0.0% | 7.0% | 12% | 5 |
| Non-matrix cycles | 17.5% | 31.9% | 17.9% | 10.3% | 21.8% | 18.7% | 20% | 6 |
| RAW stalls | 3.3% | 8.4% | 14.6% | 10.6% | 3.5% | 22.8% | 11% | 7 |
| Input data stalls | 6.1% | 8.8% | 5.1% | 2.4% | 3.4% | 0.6% | 4% | 8 |
| TeraOps/sec (92 Peak) | 12.3 | 9.7 | 3.7 | 2.8 | 86.0 | 14.1 | 21.4 | 9 |

Figure 9: TOPS figures achieved by Google TPU in different DNN architectures, compared to raw performance available.

Note the very different TOPS figures achieved for different applications and especially the difference between the figures for the two CNN implementations¹

While the TPU achieves 86.0 TOPS for the Alpha Go design (CNN0), it only achieves 14.1 TOPS for GoogLeNet (CNN1), less than 13 TOPS for the two MLP designs and less than 4 TOPS for the two LSTM designs.

This low efficiency is common across ASIC architectures. For example, the Graphcore IPU achieves a compute efficiency of only 20% during ResNet training.

GPUs suffer equally poor performance, especially when operating on shallow tensors or with small batch sizes. Comparing GoogLeNet performance on GPUs with the peak performance indicates the following:

- The V100 has a maximum efficiency of only 24.9%, falling to 11.6% for sub 2.5ms latency
- The P4 has a maximum efficiency of 45.1%, falling to 31.5% for 2.5ms latency

Omnitek DPU: FPGA-based Deep-Learning Processing Unit

¹ Norman P. Jouppi et al [*In-Datacenter Performance Analysis of a Tensor Processing Unit™*](#)

There are a wide variety of reasons for this drop off in efficiency in these GPU and ASIC architectures, including:

- Memory bandwidth limited, i.e. the DSP compute is held up waiting for SDRAM accesses
- Large parallel tensor processing block which is unable to operate efficiently on small tensors
- Overhead in distributing data to compute engines
- Thermal throttling

Using an FPGA as the underlying technology means we can rewire the architecture for different workloads to maintain optimum performance.

5.3 Adaptability to different applications

For practical applications, DNN compute acceleration is targeted to a variety of device sizes. For example, a large power device is typically viewed as the best approach for data centre acceleration while embedded and so-called 'edge' devices typically require much smaller devices.

Fortunately, FPGA vendors provide a wide variety of sizes of parts. Our FPGA-based approach means we can readily target a device suitable for the chosen application with power per device ranging from 100W down to below 10W.

Both embedded and cloud applications may require additional IP to perform the chosen operation. For example, images may need to be scaled or warped before being categorized by a CNN. The DPU's modular architecture and the flexibility of FPGAs mean that additional IP blocks can easily be accommodated in a system design. By contrast, GPUs and ASICs are limited to one or potentially only a few sizes and there is no ability to add additional compute functionality to them without significant tape-out costs.

5.4 Business Considerations: Part Cost and Time to Market

The major concerns for anyone developing any commercial system are two-fold: cost and how quickly they can get their system tested and into production.

GPU manufacturers claim GPUs are the platform of choice for quickly prototyping and testing systems designed around neural networks but FPGAs are equally strong, if not better, thanks to their superior cost efficiency and adoption of leading-edge silicon technology. For example, comparing the Arria 10 GX 1150 device with similar performance per watt GPUs or ASICs, Omnitek believe the GX 1150 FPGA offers the most cost-effective solution in terms of performance per watt per dollar.

5.5 Adopting Future Silicon Technology Nodes

The intense competition between FPGA vendors within the \$5B+ FPGA industry has ensured that FPGAs generally track the leading process nodes. Intel have recently announced the 10nm Agilex family of FPGA devices. In addition to the power, density and performance benefits due to the 10nm process, these devices contain a number of features which significantly enhance the performance of AI algorithms while also enabling their integration to other compute engines. These include:

- Support for 4 lots of 9 bit integer multiply accumulate within each DSP block, enabling performance of up to 35K MACs per device or up to 63 INT9 TOPs from DSPs alone.
- Reduced precision floating point in FP16 and bfloat16. These formats have been demonstrated to be highly efficient for training and inference and have been adopted throughout Intel technology
- Enhanced 8 input ALM coupled with 2nd generation Hyperflex routing for more efficient and higher performance logic, including low precision multiplies from 2 to 8 bit.
- Integration of Intel's CXL cache coherent interconnect and Gen5 PCI Express, enabling Agilex FPGAs to be tightly coupled to Xeon processors for AI acceleration.

6 Research Program with Oxford University

The Omnitek DPU is subject to an on-going programme of development that will extend the range of both neural network topologies that are supported and the neural network model formats with which the Omnitek DPU is able to work.

To further the development of FPGA platforms for Deep Neural Networks, Omnitek is actively promoting research into novel neural network architectures and their implementation in FPGAs with Oxford University. In particular, Omnitek is sponsoring a DPhil student, co-supervised by Oxford University and Omnitek, to carry out research related to further development and applications of the Omnitek DPU. As research yields improved topologies and optimisation techniques for AI processing, Omnitek will adapt the DPU to incorporate these developments. Some of the results of this work have already been adopted in the novel and highly efficient techniques employed in the DPU.

7 Conclusions

In this paper we have demonstrated that:

- The DPU achieves world-leading performance per watt in a mid-range programmable device and hence delivers the highest performance per watt at a given price point
- The DPU SDK supports a tool flow in which standard Python or C/C++ code written using a standard framework such as TensorFlow can be directly compiled to run on the DPU
- An FPGA-based approach provides the most flexible architecture with the ability to efficiently adapt to different applications, performance/cost/power trade-offs and future technology
- Ongoing research will be essential to maintaining a competitive edge while tracking the latest machine learning innovations
- Future generations of FPGAs, including the Intel Agilex family will lead to significant performance increases.

Omnitek believes that this highly efficient architecture – built around a flexible FPGA platform, and coupled with an extensive highly-optimised IP portfolio and an on-going research program - is the best approach to developing ICs for machine learning acceleration.

Related Papers

M.S. Abdelfattah *et al* 'DLA: Compiler and FPGA Overlay for Neural Network Inference Acceleration'
<https://arxiv.org/pdf/1807.06434.pdf> 2018

S. Ioffe & C. Szegedy 'Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift' <https://arxiv.org/pdf/1502.03167.pdf> 2015

N. P. Jouppi *et al* 'In-Datacenter Performance Analysis of a Tensor Processing Unit'
<https://arxiv.org/ftp/arxiv/papers/1704/1704.04760.pdf> 2017

S. Knowles 'Scaling Throughput Processors for Machine Intelligence'
<https://www.graphcore.ai/posts/video-scaling-throughput-processors-for-machine-intelligence> 2018

E. Nurvitadhi *et al* 'Can FPGAs Beat GPUs in Accelerating Next-Generation Deep Neural Networks?'
<http://jaewoong.org/pubs/fpga17-next-generation-dnns.pdf> 2017