The Role of FPGAs in TinyML Applications

Alexander Montgomerie-Corcoran and Christos-Savvas Bouganis.
Imperial College London, UK
am9215@ic.ac.uk

Intelligent Digital Systems Lab

Abstract

TinyML applications present a new and exciting challenge in the context of hardware acceleration, and the growing diversity of these applications afford their own unique design objectives and constraints. The existing hardware solutions for TinyML are either microcontrollers or ASIC devices. ASICs can achieve very high performance and energy efficiency, yet have limited configurability when it comes to a particular workload. Microcontrollers offer a low-cost solution to a wide range of applications, at the cost of high latency. FPGAs are highly configurable devices which allow for performance comparable to ASICs, however require a larger power budget and are often much more expense. In the context of TinyML, FPGAs can serve as a valuable platform for low-volume applications where there is a fine-grain trade-off between performance and energy. The highly-configurable aspect of FPGAs also enables efficient acceleration of other application-specific parts of the process, which is not possible with existing ML-focused ASICs.

FPGA-based ML Acceleration

The reconfigurability feature of FPGAs allows for highly customized accelerators for specific workloads, since the hardware is reusable. In the case of ML, this presents the opportunity for hardware designs specific to the FPGA's constraints.

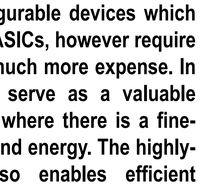
The design of FPGA-based ML accelerators generally comes under two categories:

- **Systolic Array Architecture**
- **Streaming Architecture**

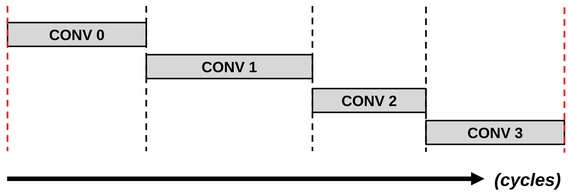
Streaming Architectures in particular are a promising architecture for highly specialized hardware designs. Figure 2 illustrates a toolflow for automating this optimization process. The toolflow is comprised of **fpgaconvnet-model** [1], **samo** [2] and **fpgaconvnet-hls** [3].

Systolic Array Architecture:

A **matrix multiplication engine**, which generalises across CNN models**. Input, weight** and **output buffers** are fed from off-chip memory. The execution of layers are **time-multiplexed.**

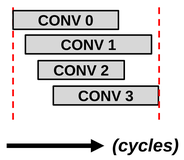


Latency = *sum of all layers*

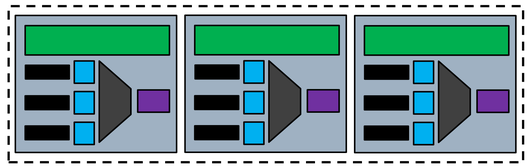


Streaming Architecture:

Follow a **deeply pipelined** principle, where layers are executed concurrently. The design of layer hardware consists of a **sliding window** feeding a **vector dot product engine.**



Latency = *max of all layers*



	Streaming	Systolic Array
Model Support	Specific	All
Deployment	Inf.	Train & Inf.
Deep Pipelining	Yes	No
Feature-Map Storage	$(K-1)WC$	$NHWC$
Weights Storage	FCK^2	FCK^2
On-Chip Memory	Low	High
Bounded by	Computation	Bandwidth

Table 1: A qualitative comparison between systolic array and streaming architectures, highlighting the benefits and drawbacks of both.

TinyML Hardware Evaluation

Hardware available for TinyML applications generally comes under three categories:

- **Microcontroller (MCU)**
- **Application-Specific Integrated Circuit (ASIC)**
- **Field Programmable Gate Array (FPGA)**

Each of these categories of hardware have different properties when it comes to performance, area and power.

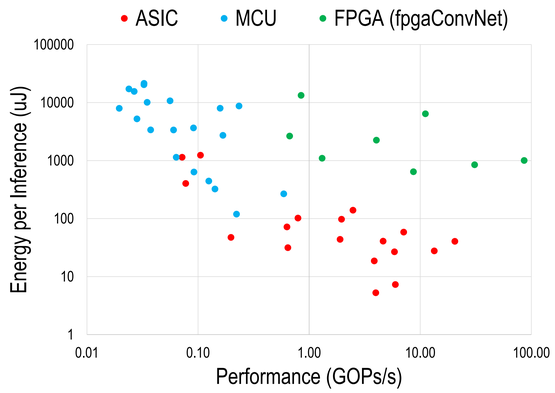


Fig. 1: Comparison of Energy and Performance results from the MLPerf Tiny benchmark [4], and preliminary results from the fpgaConvNet framework.

Model Deployment Optimisation

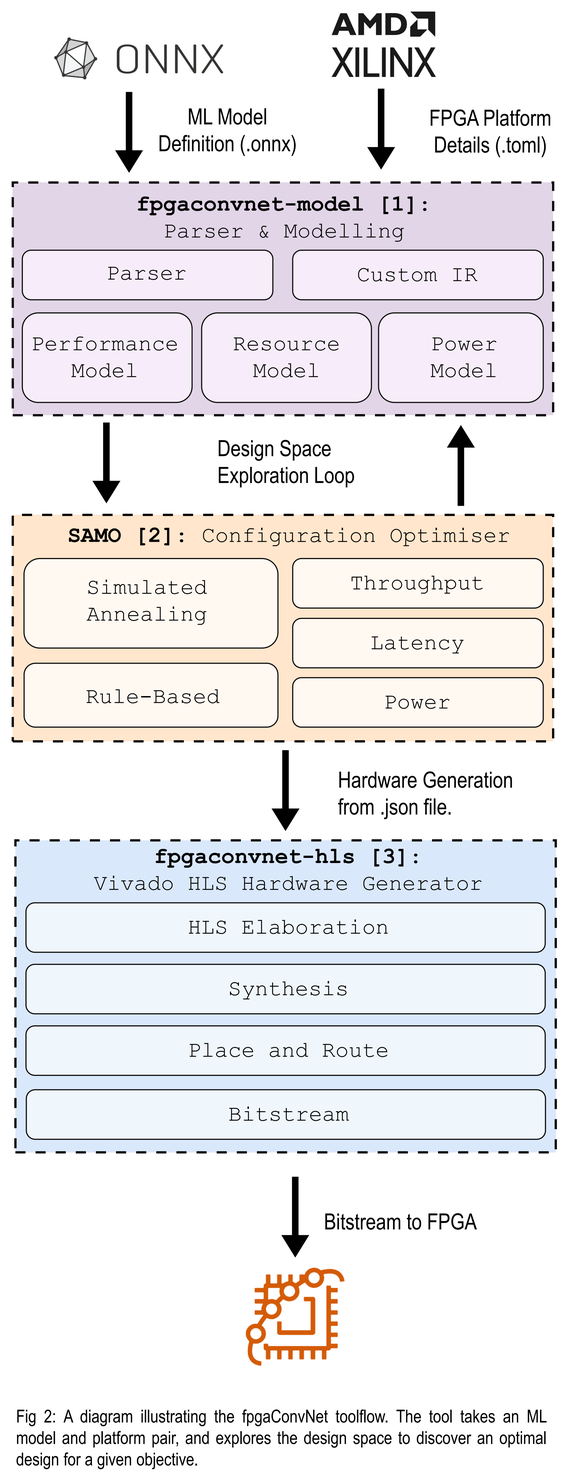


Fig 2: A diagram illustrating the fpgaConvNet toolflow. The tool takes an ML model and platform pair, and explores the design space to discover an optimal design for a given objective.

TinyML Hardware Evaluation (cont.)

From Fig. 1, it shows that FPGA devices can achieve the greatest performance, with energy consumption comparable to that of Microcontrollers. ASICs provide the lowest energy consumption overall.

However, this disparity between energy and performance for FPGA devices leads to significant power consumption, as shown in Fig. 3. The fpgaConvNet tool draws factors higher power consumption compared to the other platforms.

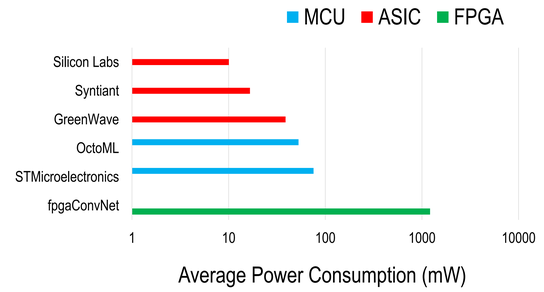


Fig 3: Comparison of average system power from the submitters of the MLPerf Tiny benchmark [4], as well as preliminary results from the fpgaConvNet framework.

Conclusion

Overall, FPGA devices present a new alternative platform for TinyML applications. A comparison between all the available categories of devices is given in Table 2. FPGAs can achieve really high performance at the cost of higher power compared to any other device. Furthermore, they are highly configurable, which benefits adaptable use cases, and allows for acceleration of other aspects of the inference pipeline. The main drawbacks surround the cost of FPGAs as well as their high power consumption. This makes them less suitable for high-volume and power-constrained applications.

	MCU	ASIC	FPGA
Performance	Low	High	High
Power	Low	Low	High
Energy	Medium	Low	Medium
Cost	Low	Mediuim	High
Configurability	High	Low	High

Tab 2: General comparison of Microcontrollers, ASICs and FPGAs in the context of TinyML applications.

References

[1] https://github.com/AlexMontgomerie/fpgaconvnet-model
[2] https://github.com/AlexMontgomerie/samo
[3] https://github.com/AlexMontgomerie/fpgaconvnet-hls
[4] https://mlcommons.org/en/inference-tiny-10//