


XNORBIN: A 95 TOp/s/W hardware accelerator for binary convolutional neural networks

Conference Paper**Author(s):**

Bahou, Andrawes A.; Karunaratne, Geethan; Andri, Renzo; [Cavigelli, Lukas Arno Jakob](#) ; [Benini, Luca](#) 

Publication date:

2018

Permanent link:

<https://doi.org/10.3929/ethz-b-000329926>

Rights / license:

[In Copyright - Non-Commercial Use Permitted](#)

Originally published in:

<https://doi.org/10.1109/CoolChips.2018.8373076>

XNORBIN: A 95 TOP/s/W Hardware Accelerator for Binary Convolutional Neural Networks

Andrawes Al Bahou*, Geethan Karunaratne*, Renzo Andri, Lukas Cavigelli, Luca Benini
Integrated Systems Laboratory, ETH Zurich, Zurich, Switzerland

*These authors contributed equally to this work and are listed in alphabetical order.

Abstract

Deploying state-of-the-art CNNs requires power-hungry processors and off-chip memory. This precludes the implementation of CNNs in low-power embedded systems. Recent research shows CNNs sustain extreme quantization, binarizing their weights and intermediate feature maps, thereby saving $8\text{-}32\times$ memory and collapsing energy-intensive sum-of-products into *XNOR*-and-*popcount* operations.

We present XNORBIN, a flexible accelerator for binary CNNs with computation tightly coupled to memory for aggressive data reuse supporting even non-trivial network topologies with large feature map volumes. Implemented in UMC 65nm technology XNORBIN achieves an energy efficiency of 95 TOP/s/W and an area efficiency of 2.0 TOP/s/MGE at 0.8 V.

I. INTRODUCTION & RELATED WORK

The recent success of convolutional neural networks (CNNs) have turned them into the go-to approach for many complex machine learning tasks. Computing the forward pass of a state-of-the-art image classification CNN requires ≈ 20 GOP/frame (1 multiply-accumulate corresponds to 2 Op) and access to 20M-100M weights [1]. Such networks are out of reach for low-power (mW-level) embedded systems and are typically run on W-level embedded platforms or workstations with powerful GPUs. Hardware accelerators are essential to push CNNs to mW-range low-power platforms, where state-of-the-art energy efficiency at negligible accuracy loss is achieved using binary weight networks [2]. For maximum energy efficiency, it is imperative to store repeatedly accessed data on-chip and limit any off-chip communication. This puts stringent constraints on the CNNs (weights and intermediate feature map size) that can fit the device, limiting applications and crippling accuracy. Binary neural networks (BNNs) use bipolar binarization (+1/-1) for both the model weights and feature maps, reducing overall memory size and bandwidth constraints by $\approx 32\times$, and simplifying the costly sum-of-products computation to mere XNOR-and-popcount operations while keeping an acceptable accuracy penalty relative to full-precision networks [3], also thanks to iterative improvement through multiple BNN convolutions [4].

Recently, a few energy-efficient hardware accelerators for BNNs have been presented, but they rely on analog integration [5] or analog in-memory optimization [6] and severely constrain the network size. We present XNORBIN, a fully-digital hardware accelerator targeting a wide range of state-of-the-art fully-binary CNNs to address both the memory and computation energy challenges for large-scale computer vision tasks like object recognition on ImageNet. The key operations of BNNs are 2D-convolutions of multiple binary (+1/-1) input feature maps and binary (+1/-1) filter kernel sets, resulting in multiple integer-valued feature maps. This convolutions can be formulated as many parallel XNOR-and-popcount operations with the potential for intensive data reuse. The activation function and the optional batch normalization can then be collapsed to a re-binarization on a pre-computed per-output feature map threshold value and can be applied on-the-fly. An optional pooling operation can be enabled after the re-binarization. XNORBIN is complete in the sense that it implements all common BNN operations (e.g. those of the binary AlexNet described in [3]) and supports a wide range of hyperparameters. Furthermore, we provide a tool for automatic mapping of trained BNNs from the PyTorch deep learning framework to a control stream for XNORBIN.

II. ARCHITECTURE

A top-level overview of the chip architecture is provided in Fig. 1. The XNORBIN accelerator sequentially processes each layer of the BNN.

Data flow: To support up to 7×7 kernel sizes, the processing core of XNORBIN is composed of an array (cf. Fig. 3) of 7 BPUs (Basic Processing Units), where every BPU includes a set of 7 `xnor_sum` units. These units calculate the XNOR-and-popcount result on 16 bit vectors, containing values of 16 feature maps at a specific pixel. The outputs of all 7 `xnor_sum` units in a BPU are added-up, computing one output value of a 1D convolution on an image row each cycle. On the next higher level of hierarchy, the results of the BPUs are added up to produce one output value of a 2D convolution (cf. Fig. 3). Cycle-by-cycle, a convolution window slides horizontally over the image. The resulting integer value is forwarded to the DMA controller, which includes a near-memory compute unit (CU). The CU accumulates the partial results by means of a read-add-write operation, since the feature maps are processed in tiles of 16. After the final accumulation of partial results, the unit also performs the thresholding/re-binarization operation (i.e. activation and batch normalization). When binary results have to be written back to memory, the DMA also handles packing them into 16 bit words.

Data re-use/buffering: XNORBIN comes with three levels of memory and data buffering hierarchy. 1) At the highest level, the

main memory stores the feature maps and the partial sums of the convolutions. This memory is divided into two SRAM blocks, where one serves as the data source (i.e. contains the current layer’s input feature maps) while the other serves as data sink (i.e. contains the current output feature maps/partial results), and switching their roles as layer changes. These memories are sized such that they can store the worst-case layer of a non-trivial network (implemented: 128 kbit and 256 kbit to accommodate binary AlexNet), thus avoid the tremendous energy cost of pushing intermediate results off-chip. Additionally, a *parameter buffer* stores the weights, binarization threshold, and configuration parameters and is sized to fit the target network, but may be used as a cache for an external flash memory storing these parameters. Integrating this accelerator with a camera sensor would allow to completely eliminate off-chip communication. 2) On the next lower level of hierarchy, the row banks are used to buffer rows of the input feature maps to relieve pressure on the SRAM. Since these row banks need to be rotated when shifting the convolution window down, they are connected to the BPU array through a crossbar. 3) The crossbar connects to the working memory inside the BPUs, the *controlled shift registers* (CSRs, cf. Fig. 3), for the input feature maps and the filter weights. These are shifted when moving the convolution window forward. All the data words in the CSRs are accessible in parallel and steered to the `xnor_sum` units.

Programmability: XNORBIN supports CNNs of arbitrary depths by streaming the network parameters from an external memory. However, the size of the BNN’s layer with the largest pair of input and output feature maps has to fit into the main memory (i.e. 250 kbit for the actual implementation of XNORBIN). The succession of CNN layers is configurable. XNORBIN supports adjustable feature map dimensions (height, width, channel length), as long as the volume of the largest intermediate feature map fits in main memory. It can handle convolution windows of up to 7×7 and configurable stride. Furthermore, any convolution layer with a filter size larger than 7×7 would need to be split into smaller convolutions due to the number of parallel working BPUs, `xnor_sum` units per BPU, the number of row banks, and the size of the CSRs, thereby introducing a large overhead.

III. IMPLEMENTATION RESULTS

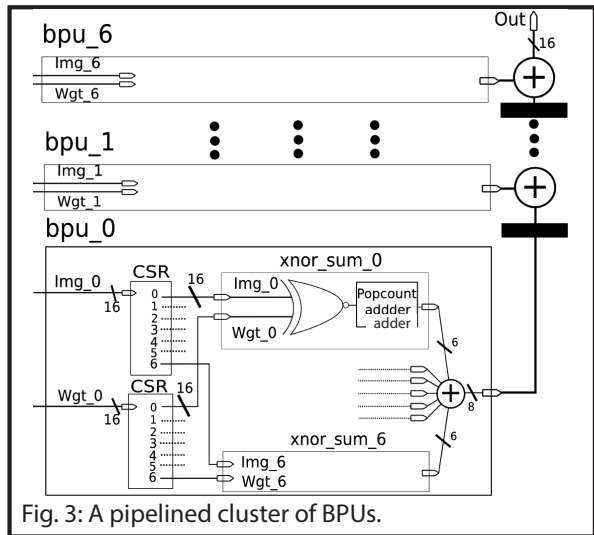
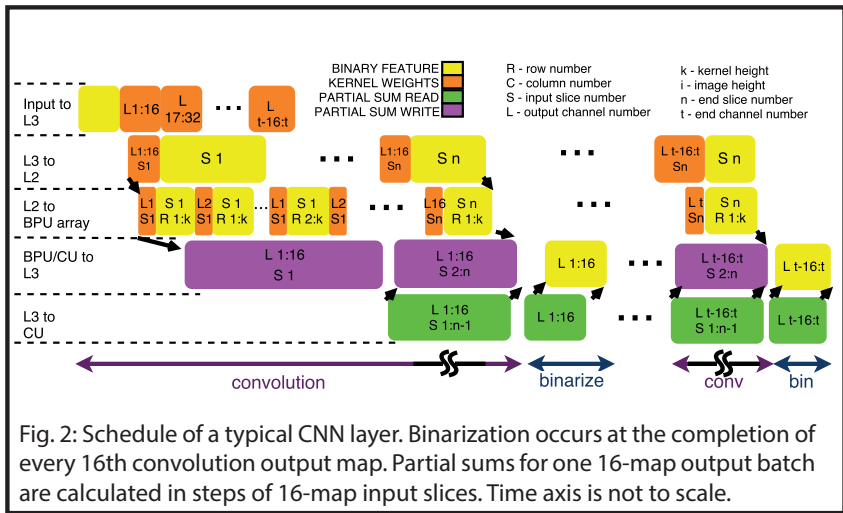
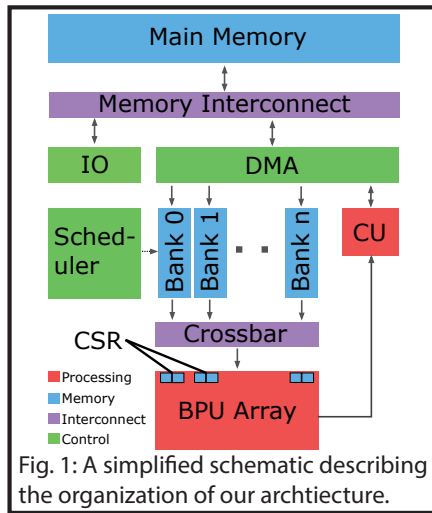
The design has been implemented in UMC 65nm technology. Evaluations for the typical-case corner at 25 °C for 1.2V and 0.8V, have yielded a throughput of 746 GOp/s and 244 GOp/s and an energy efficiency of 23.3 TOP/s/W and 95 TOP/s/W, respectively. The system consumes 1.8 mW@0.8V from which 69% are in the memory, 14.6% in the DMA and crossbar and 13% in the BPUs. The key performance and physical characteristics are presented in Fig. 4. The implementation parameters such as memory sizes have been chosen to support BNN models up to the size of binary AlexNet. A binary AlexNet pretrained on ImageNet [3] has been used for evaluation purposes. The throughput and energy consumption per layer are shown in Tbl. 1. The results are implicitly bit-true—there is no implementation loss such as going from fp32 to a fixed-point representation, since all intermediate results are integer-valued or binary. We compare energy efficiency of XNORBIN to state-of-the-art CNN accelerators in Tbl. 3. To the best of our knowledge, this is the first hardware accelerator for binary neural networks. The closest comparison point are the FPGA-based FINN results [7] with a $168 \times$ higher energy consumption when running BNNs. The strongest competitor is [2], which is a binary-weight CNN accelerator strongly limited by I/O energy, requiring $25 \times$ more energy per operation than XNORBIN.

IV. CONCLUSION

Thanks to the binarization of the neural networks, the memory footprint of the intermediate results as well as the filter weights could be reduced by $8\text{-}32 \times$, making XNORBIN capable to fit all intermediate results of a simple, but realistic BNN, such as binary AlexNet, into on-chip memory with a mere total accelerator size of 0.54 mm^2 . Furthermore, the computational complexity decreases significantly as full-precision multiplier-accumulate units are replaced by XNOR and pop-count operations. Due to these benefits—smaller compute logic, keeping intermediate results on-chip, reduced model size—XNORBIN outperforms the overall energy efficiency of existing accelerators by more than $25 \times$.

REFERENCES

- [1] A. Canziani, A. Paszke, and E. Culurciello, “An analysis of deep neural network models for practical applications,” *CoRR*, vol. abs/1605.07678, 2016.
- [2] R. Andri, L. Cavigelli, D. Rossi, and L. Benini, “YodaNN: An architecture for ultralow power binary-weight cnn acceleration,” *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)*, vol. 37, no. 1, pp. 48–60, 2018.
- [3] M. Rastegari, V. Ordonez, J. Redmon, and A. Farhadi, “Xnor-net: Imagenet classification using binary convolutional neural networks,” in *Proc. European Conference on Computer Vision*, 2016, pp. 525–542.
- [4] X. Lin, C. Zhao, and W. Pan, “Towards accurate binary convolutional neural network,” in *Advances in Neural Information Processing Systems (NIPS)*, 2017, pp. 345–353.
- [5] D. Bankman, L. Yang, B. Moons, M. Verhelst, and B. Murmann, “An always-on $3.8 \mu\text{j}/86\%$ cifar-10 mixed-signal binary cnn processor with all memory on chip in 28nm cmos,” in *Proc. IEEE International Solid-State Circuits Conference (ISSCC)*, 2018, pp. 222–224.
- [6] X. Sun, X. Peng, P.-Y. Chen, R. Liu, J.-s. Seo, and S. Yu, “Fully parallel rram synaptic array for implementing binary neural network with (+1,-1) weights and (+1,0) neurons,” in *Proc. IEEE Asia and South Pacific Design Automation Conference (ASP-DAC)*, 2018, pp. 574–579.
- [7] Y. Umuroglu, N. J. Fraser, G. Gambardella, M. Blott, P. Leong, M. Jahre, and K. Vissers, “Finn: A framework for fast, scalable binarized neural network inference,” in *Proc. ACM International Symposium on Field-Programmable Gate Arrays*, 2017, pp. 65–74.



Physical Characteristics

Technology	UMC 65 nm, 8 metal Layers
Package	QFN-56
# Pads	40 (i: 18, o:6, clk/test: 6, pwr: 8)
Core Area	0.54 mm ²
Circuit Complexity	368 kGE
On-chip SRAM	430 kbit

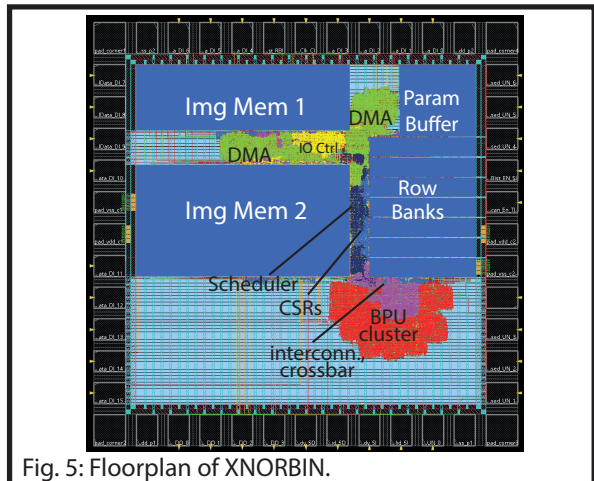
Performance & Efficiency @ 1.2V

Max Clock Frequency	core: 476 MHz, io: 238 MHz
Power	32 mW (core) + 24 mW (pad)
Peak Throughput	746 GOp/s
Core Power-Efficiency	23324 GOp/s/W @ 1.2 V
Device Power-Efficiency	13270 GOp/s/W @ 1.2 V
FPS AlexNet binary layers	20.8 fps

Performance & Efficiency @ 0.8V

Max Clock Frequency	core: 156 MHz, io: 78 MHz
Power	1.97 mW (core) + 7.9 mW (pad)
Peak Throughput	244 GOp/s
Core Power-Efficiency	94800 GOp/s/W @ 0.8 V
Device Power-Efficiency	23200 GOp/s/W @ 0.8 V
FPS AlexNet binary layers	6.81 fps

Fig. 4: Key figures of XNORBIN.



Layer	Operation	Number of Basic Ops	Number of Cycles	I/O Energy [J]	Core Energy 1.2V [J]	Core Energy 0.8V [J]
1	Conv	448M	2.10M	13.7μ	151μ	36.6μ
	Pool	173k	93.3k			
	Binarize	43.3k	21.6k			
2	Conv	150M	1.85M	17.7μ	127μ	30.6μ
	Binarize	13.8k	6.91k			
	Conv	224M	2.77M			
3	Binarize	13.8k	6.91k	26.5μ	190μ	45.9μ
	Conv	150M	2.40M			
	Pool	36.9k	21.6k			
4	Binarize	9.22k	4.61k	17.7μ	166μ	40.1μ
	Conv	37.7M	9.44M			
	Binarize	4.10k	2.05k			
5	Conv	37.7M	9.44M	755μ	646μ	156μ
	Binarize	4.10k	2.05k			
6	Conv	16.8M	4.19M	336μ	287μ	69.4μ

Tbl. 1: Layer-wise performance on AlexNet.

	CSRs	Row Banks	Img Mem1	Img Mem2	Param Buffer
Mem Size	1.5 kbit	28 kbit	128 kbit	256 kbit	16 kbit
Data width	16 bit	16 bit	32 bit	32 bit	32 bit
Type	register	2-port regfile	1-port SRAM	1-port SRAM	2-port regfile
# Rows	7	256	4096	8192	512
Peak rd/cycle	7	1	0.5	0.5	0.5
Peak wr/cycle	1	0.14	0.5	0.5	0.5

Tbl. 2: Storage elements in memory hierarchy.

Design	Power [mW]	Efficiency [GOp/s/W]	Freq. [MHz]	Core Area [mm ²]	Process
FINN (FPGA)	2.3k	567	200	-	Z-7045
NeuFlow (FPGA)	10k	15	-	-	-
NeuFlow	600	490	400	13	IBM45
Eyeriss	278	246	250	16	TSMC65
ShiDianNao	320	400	1000	5	TSMC65
EIE	590	5000	800	41	TSMC45
Origami (@0.8V)	core: 93 pads: 144	core: 803 device: 220	189	3.09	UMC65
YodaNN (@1.2V)	core: 39 pads: 395	core: 9610 device: 870	480	1.91	UMC65
YodaNN (@0.6V)	core: 0.26 pads: 15.54	core: 61k device: 980	27.5	1.91	UMC65
XNORBIN (@1.2V)	core: 32 pads: 24.2	core: 23k device: 13k	476	0.54	UMC65
XNORBIN (@0.8V)	core: 2.6 pads: 7.9	core: 95k device: 23k	156	0.54	UMC65

Tbl. 3: Comparison of various state-of-the-art accelerators.