TileNET: Scalable Architecture for High-throughput Ternary Convolution Neural Networks using FPGAs

By

Sahu Sai Vikram (BL.EN.P2VLD15024)

Under the Guidance of Madhura. P(Associate Professor)

Contents

MATA AMRITANANDAMAYI MATH

श्रद्धावान् लभते ज्ञानम्

Abstract

- Advanced driver assistance systems (ADAS) necessitates high-throughput in the order of about few 10's of TeraMACs per second (TMACS).
- In addition to throughput, accuracy is also of very high importance. It has been observed that **ternarization of input features results in minimal loss of accuracy**.
- The work presents a novel tiled architecture for CNNs with ternarized weights.

CNN(Convolutional Neural Network)

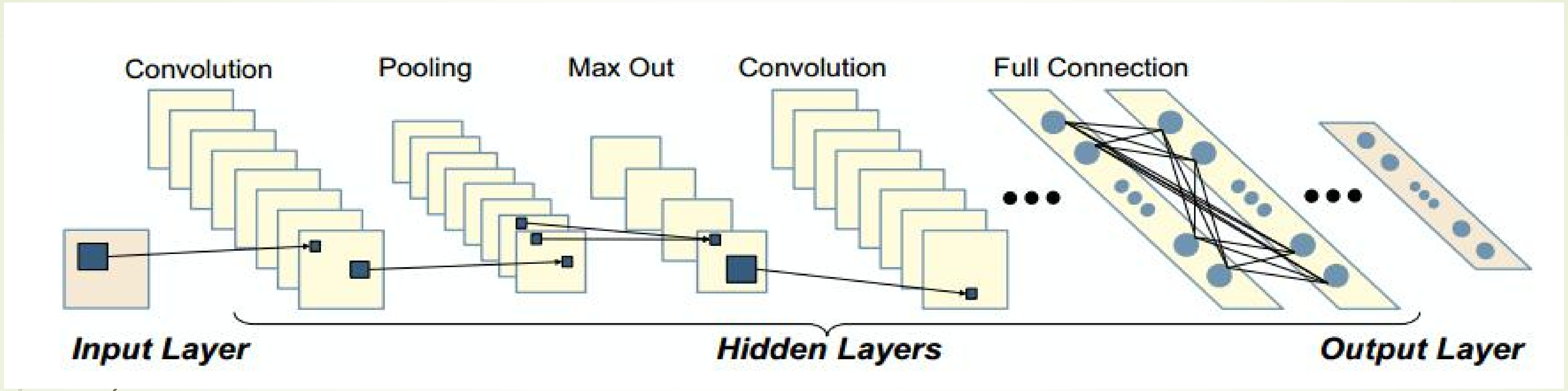


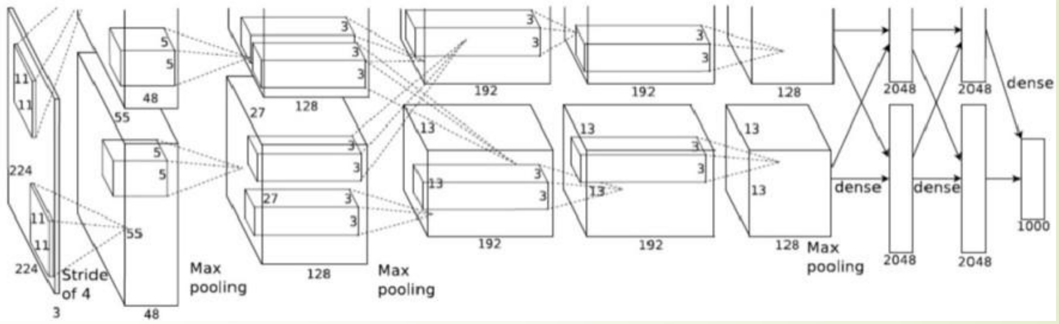
Fig: Structure of the CNN(Ref[3])

CNN

- CNN is a multi-layered neural network with multiple hidden layers.
- **Input layer**
- **Output layer**
- **Hidden layers**
  - **Convolution layer**
  - **ReLU layer:** $f(x) = \max(0,x)$
  - **Max Pooling layer**
  - **Full-connected layers**

AlexNet

- Alexnet is a Deep Convolutional Neural Network for image classification that won the ILSVRC-2012 competition(Ref [6])
- AlexNet has **5 convolutional layers, 3 sub sampling layers, 3 fully connected layers**



Architecture for AlexNet

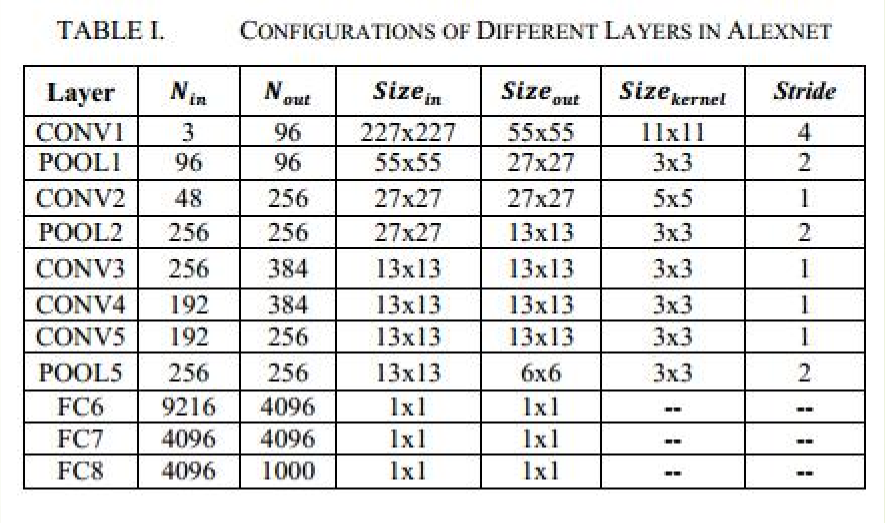


TABLE I. CONFIGURATIONS OF DIFFERENT LAYERS IN ALEXNET

Layer	N_{in}	N_{out}	$Size_{in}$	$Size_{out}$	$Size_{kernel}$	$Stride$
CONV1	3	96	227x227	55x55	11x11	4
POOL1	96	96	55x55	27x27	3x3	2
CONV2	48	256	27x27	27x27	5x5	1
POOL2	256	256	27x27	13x13	3x3	2
CONV3	256	384	13x13	13x13	3x3	1
CONV4	192	384	13x13	13x13	3x3	1
CONV5	192	256	13x13	13x13	3x3	1
POOL5	256	256	13x13	6x6	3x3	2
FC6	9216	4096	1x1	1x1	--	--
FC7	4096	4096	1x1	1x1	--	--
FC8	4096	1000	1x1	1x1	--	--

Literature Survey

DRRA(Dynamically Reconfigurable Resource Array)

It consists of:

1. Register Files
2. Morphable Datapath Unit
3. Circuit Switched Blocks
4. Sequencer

Synthesized in 65nm technology with Frequency of 500MHz

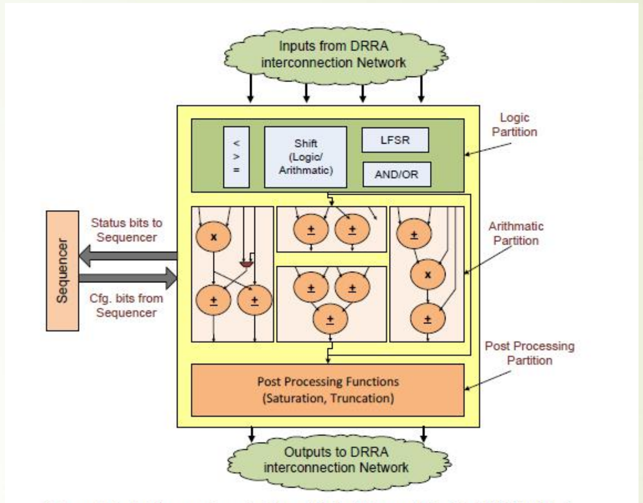

Inputs from DRRA interconnection Network
Logic Partition
Shift (Logic/ Arithmatic)
LFSR
AND/OR
Status bits to Sequencer
Sequencer
Cfg. bits from Sequencer
Arithmatic Partition
Post Processing Partition
Post Processing Functions (Saturation, Truncation)
Outputs to DRRA interconnection Network

Figure 4.5: A illustration of a Data-Path Unit used in the DRRA fabric.

DRRA (ref [1])

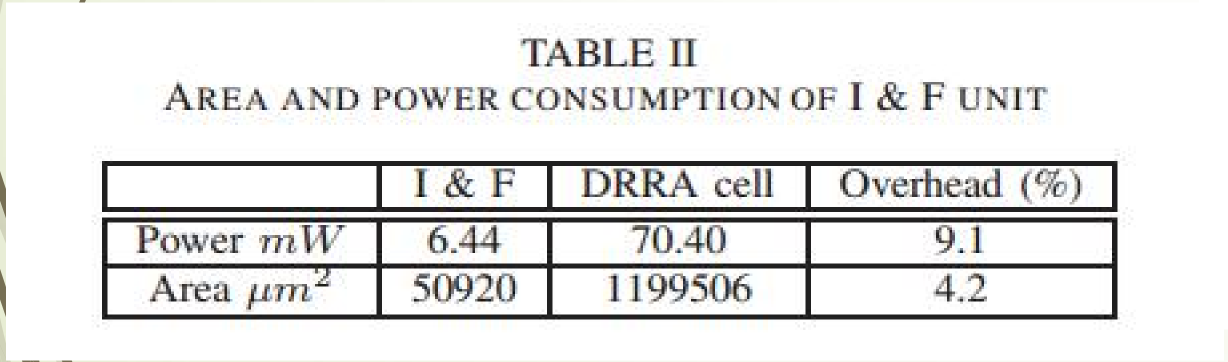
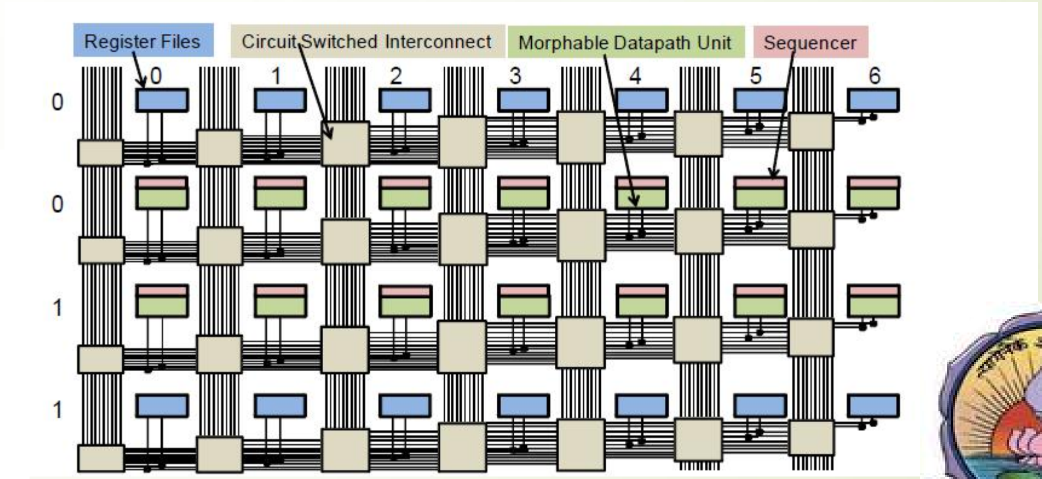


TABLE II
AREA AND POWER CONSUMPTION OF I & F UNIT

	I & F	DRRA cell	Overhead (%)
Power mW	6.44	70.40	9.1
Area μm^2	50920	1199506	4.2



Literature Survey

EMAX(Energy Efficient Multimode Accelerator)

PE has
- 2 EUs
- EAG
- LMM
- Constant Registers

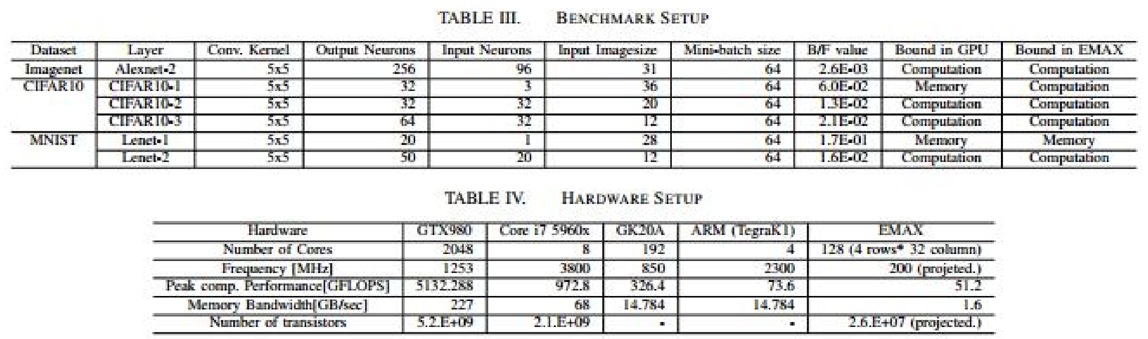
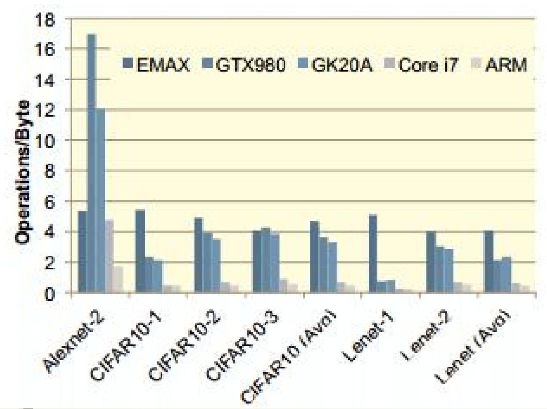


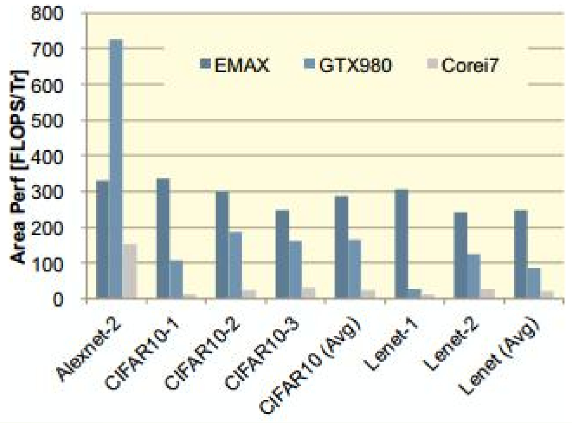
TABLE III. BENCHMARK SETUP

Dataset	Layer	Conv. Kernel	Output Neurons	Input Neurons	Input Imagesize	Mini-batch size	B/F value	Bound in GPU	Bound in EMAX
Imagenet	Alexnet-2	5x5	256	96	31	64	2.6E-03	Computation	Computation
CIFAR10	CIFAR10-1	5x5	32	3	36	64	6.0E-02	Memory	Computation
	CIFAR10-2	5x5	32	32	20	64	1.3E-02	Computation	Computation
	CIFAR10-3	5x5	64	32	12	64	2.1E-02	Computation	Computation
MNIST	Lenet-1	5x5	20	1	28	64	1.7E-01	Memory	Memory
	Lenet-2	5x5	50	20	12	64	1.6E-02	Computation	Computation

TABLE IV. HARDWARE SETUP

Hardware	GTX980	Core i7 5960x	GK20A	ARM (TegraK1)	EMAX
Number of Cores	2048	8	192	4	128 (4 rows* 32 column)
Frequency [MHz]	1253	3800	850	2300	200 (projeted.)
Peak comp. Performance[GFLOPS]	5132.288	972.8	326.4	73.6	51.2
Memory Bandwidth[GB/sec]	227	68	14.784	14.784	1.6
Number of transistors	5.2.E+09	2.1.E+09	-	-	2.6.E+07 (projected.)





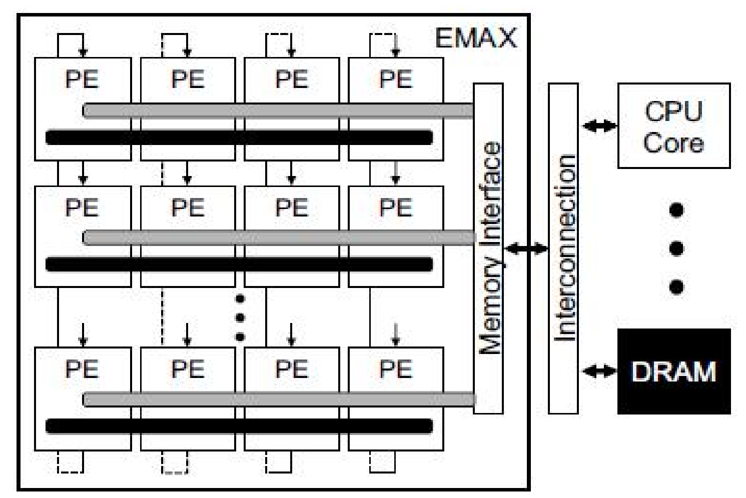


Fig. 6. EMAX Architecture

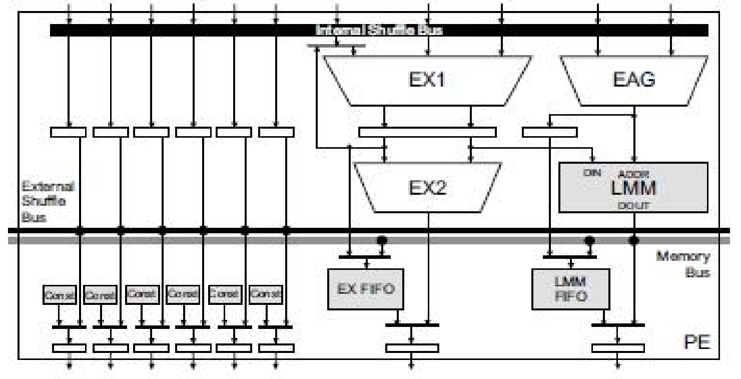


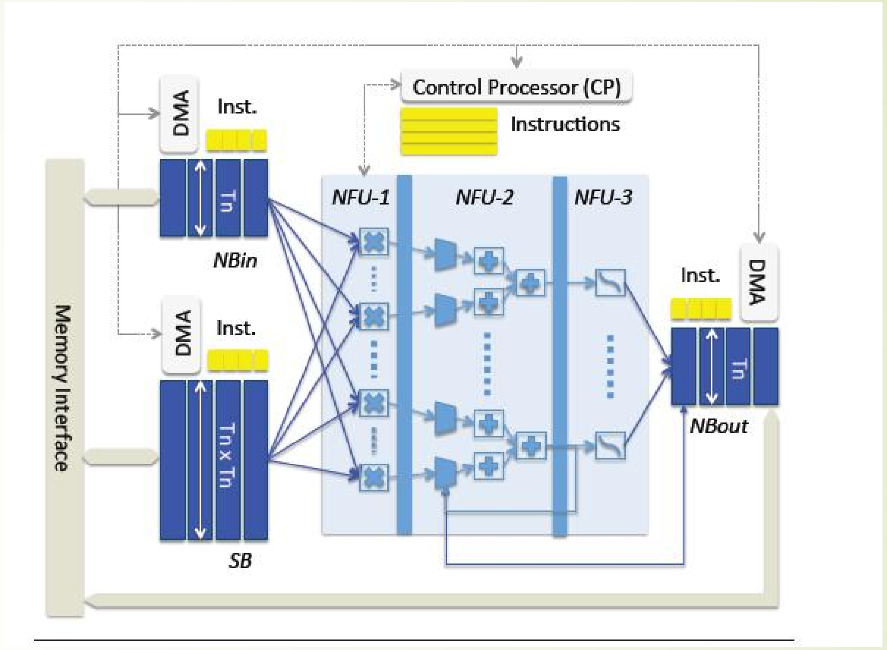
Fig. 7. PE Microarchitecture

EMAX(ref [3])

Literature Survey

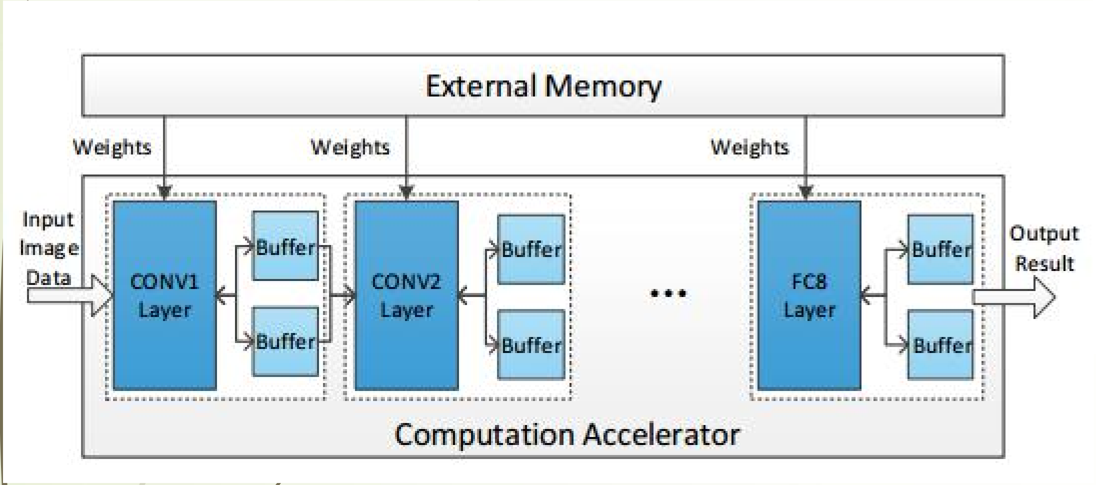
DianNao

- Throughput of 452 GOP/s in a small footprint of 3.02 mm2 and 485 mW.
- Compared to a 128-bit 2GHz SIMD processor, the accelerator is 117.87x faster, reduce the total energy by 21.08x.
- The accelerator characteristics are obtained after layout at 65nm



DianNao(ref [2])

Literature Survey



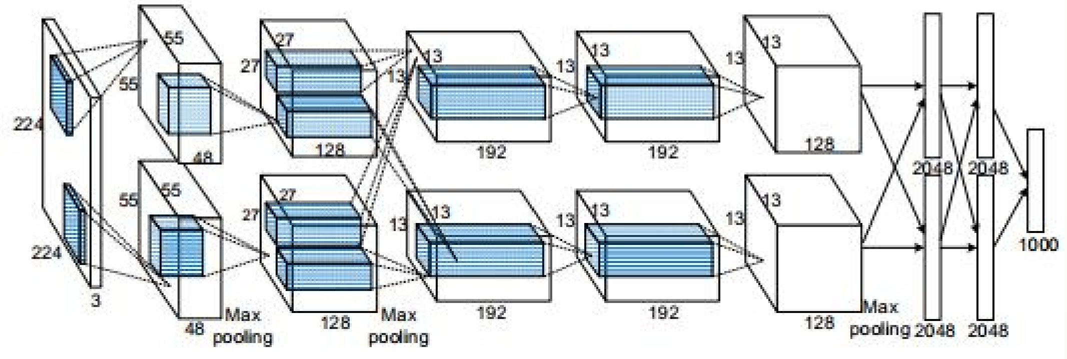


Fig.1. An overview of the large-scale CNN model AlexNet.

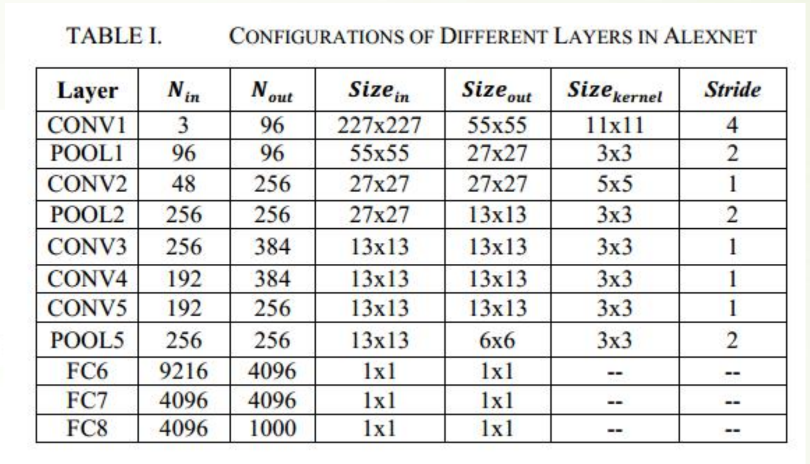


TABLE I. CONFIGURATIONS OF DIFFERENT LAYERS IN ALEXNET

Layer	N_{in}	N_{out}	$Size_{in}$	$Size_{out}$	$Size_{kernel}$	*Stride*
CONV1	3	96	227x227	55x55	11x11	4
POOL1	96	96	55x55	27x27	3x3	2
CONV2	48	256	27x27	27x27	5x5	1
POOL2	256	256	27x27	13x13	3x3	2
CONV3	256	384	13x13	13x13	3x3	1
CONV4	192	384	13x13	13x13	3x3	1
CONV5	192	256	13x13	13x13	3x3	1
POOL5	256	256	13x13	6x6	3x3	2
FC6	9216	4096	1x1	1x1	--	--
FC7	4096	4096	1x1	1x1	--	--
FC8	4096	1000	1x1	1x1	--	--

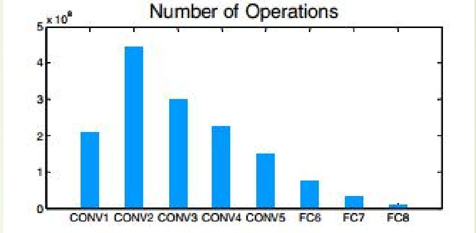


Fig.2. Number of operations for different layers in AlexNet.

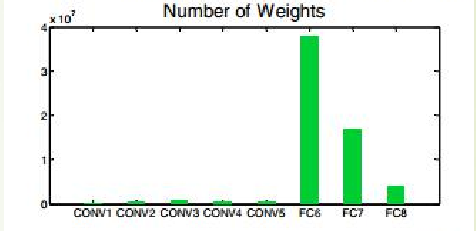


Fig.3. Number of weights for different layers in AlexNet.

Ref [4]

A high performance FPGA-based accelerator for large-scale convolutional neural networks

Ternary Weights

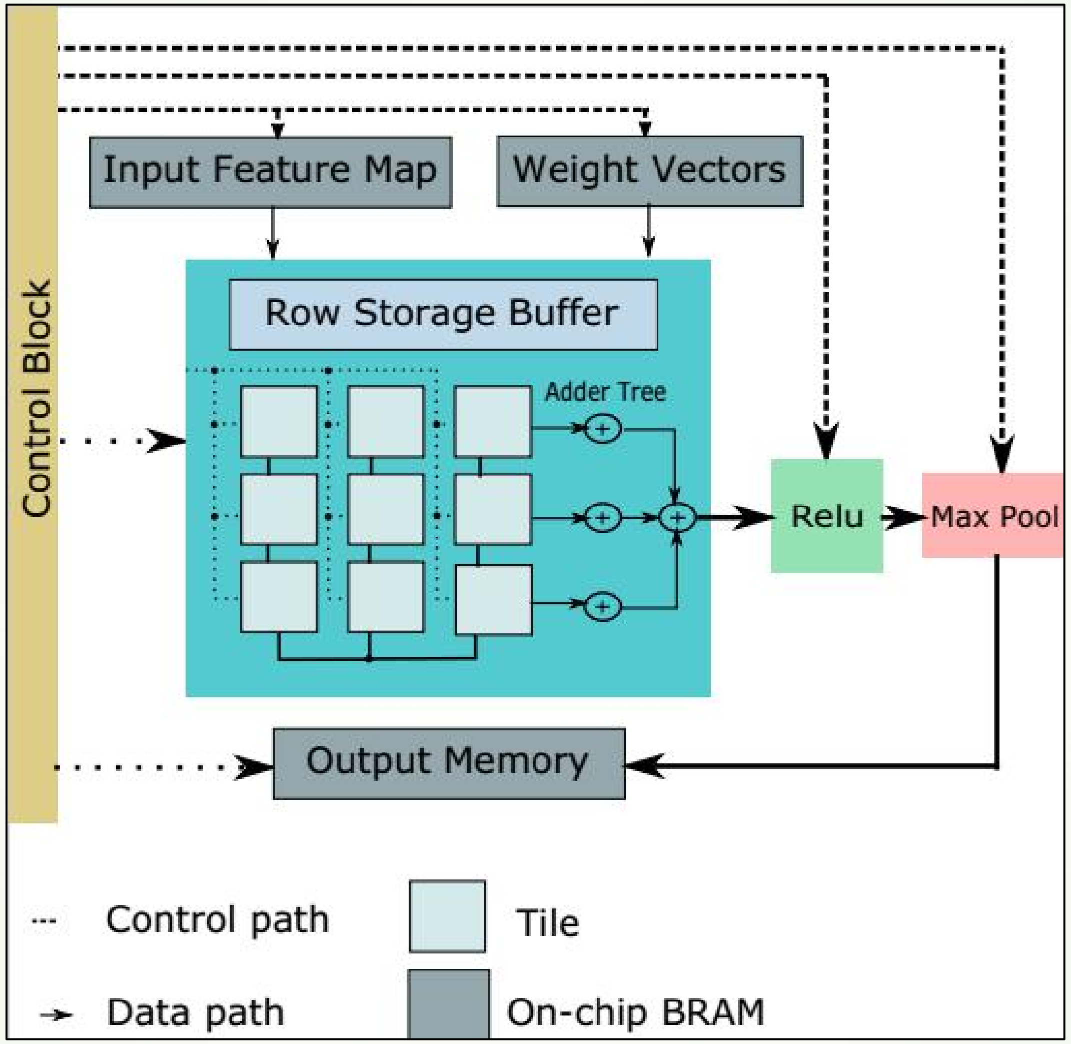
CNN	Weight	Activation	Scheme	Accuracy(MNIST)
Binary Connect	-1,1	Full Precision	Deterministic Binarization	98.82%
Ternary WN	-1,0,1	Full Precision	Optimization with threshold-based Ternarization function	99.35%

Ref [5]: Impact of Binary & Ternary Weights on Accuracy

Proposed Architecture

- Considerations:
  - A generic and scalable tiled architecture for ternary CNNs that is independent of the model used
  - To Achieve a high Throughoutput using Ternarization of weights

TileNet



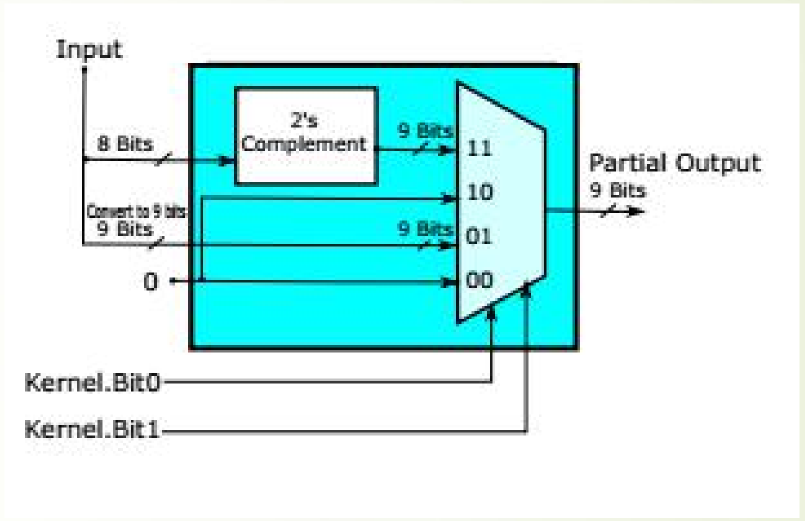
TileNet Architecture

Ternary Multiplier

- 4x1 multiplexer with a 2's complement computation.
- As a result, the multiplier with ternarized weight vector has significantly lower resource requirements as compared to an 8-bit fixed-point multiplier.

Weight vector(w)	partial Output(y)
00	0
01	p
10	0
11	-p

y = w x p

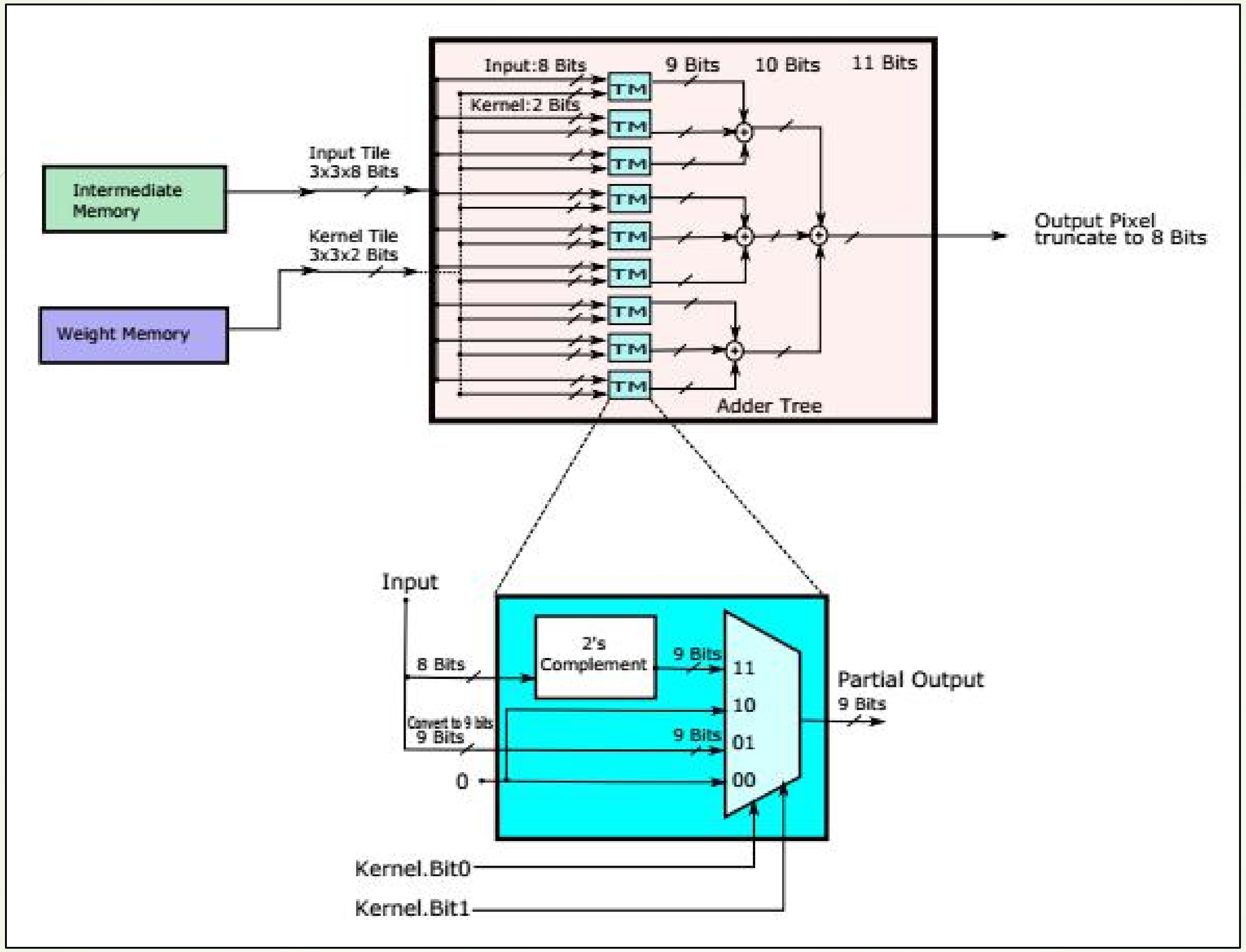


Ternary Multiplier

Processing Element(PE)

- The Processing Element does the computation of the entire Tile
- The Tile size is considered as 3x3.
- A single tile consists of 9 Ternary multipliers followed by 3- input adder tree. This forms a PE.
- Multiple such tiles are instantiated in parallel for computing a part of a convolution layer.
  - PExNxC
- The Convolution layer is followed by Rectified Linear Unit (ReLu) and Max Pooling.

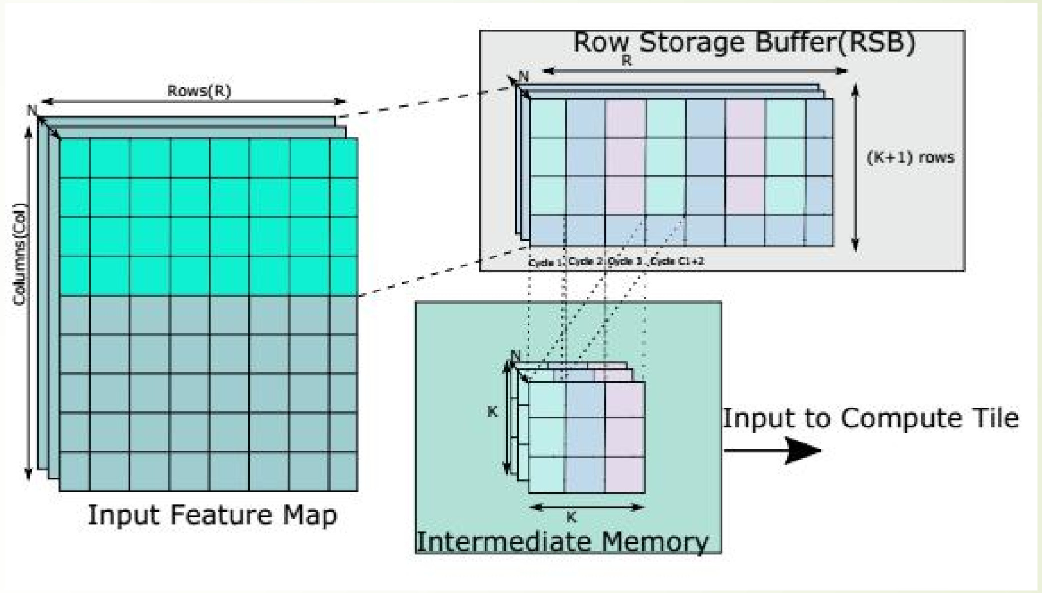
Processing Element



Processing Element

Memory Organisation

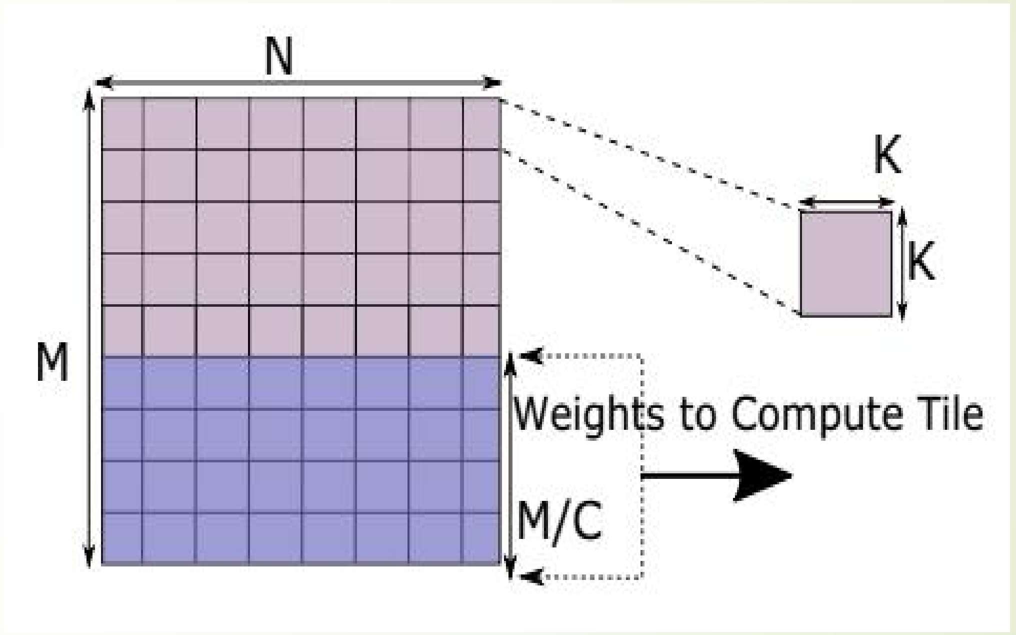
- The input data from the External Memory is loaded to Row Storage Buffer(RSB) which can store (K+1) rows.
- The Intermediate memory, holds the data required for the computation.
- The data is written across the column for the intermediate memory, when new data comes, it will flush out the data from the Left side.



Input Memory

Memory Organisation

- The Kernel data is stored in internal memory.
- The each location can store a single kernel vector



Weight Memory

Implementation

- The implementation is done on the FPGA using Xilinx Vivado Tool
  - Implemented a PE on the different devices like Virtex, Kintex, Zinq and Artex to check the performance on these devices
  - Implemented the Convolutional Layers of the AlexNet on Virtex 7 FPGA.

Implementation

Synthesized Results of a PE on different Devices is as follows.

DEVICE	LUTS	FFs	Delay (ns)	Freq (MHz)	Power(W)
xc7vx1140 (Virtex)	176	128	1.801	555.24	0.71
xc7k480 (Kintex)	176	128	1.807	553.403	0.21
xc7z100 (Zinq)	176	128	1.808	553	0.324
xc7a200 (Artix)	176	128	3.426	291.886	0.193

Implementation

The results are synthesized in Vertex 7

Device	xc7vx1140(Vertex 7)
LUTS	712000
FFS	1424000
BRAMS	1880

Parameters:

N	- input feature Maps
M	- output feature Maps(No. of kernel Weights)
k	- kernel width
Nt	- No. of tiles in Kernel
Rin,Cin	- input feature map width and height
Ro,Co	- output feature map width and height
C	- Parallelization factor
M/c	- No. of times layer has to run

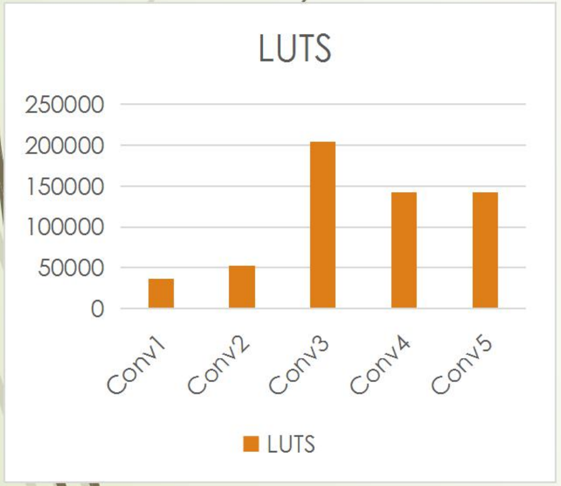
Layer	N	M	K	Nt	Rin	Cin	Ro	Co	M/C	C value	Max Pooling	Ternary Compute	LUTS
Conv1	3	96	11x11	16	224	224	55	55	16	6	3x3	2592	36014
Conv2	48	256	5x5	4	55	55	27	27	128	2	3X3	3456	52217
Conv3	256	384	3x3	1	13	13	13	13	64	6	NA	13824	204038
Conv4	192	384	3x3	1	13	13	13	13	64	6	NA	10368	142668
Conv5	192	256	3x3	1	13	13	13	13	64	4	3X3	6912	142368
Total												**37152**	**577305**

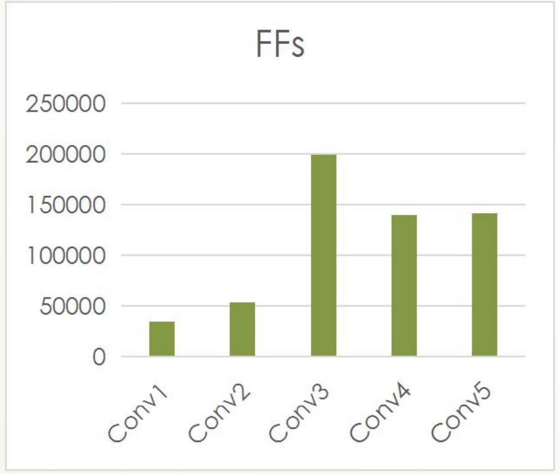
%LUTS	**80.08**
Freq	**453.30 MHz**
Throughput	**33.68 TOPs**

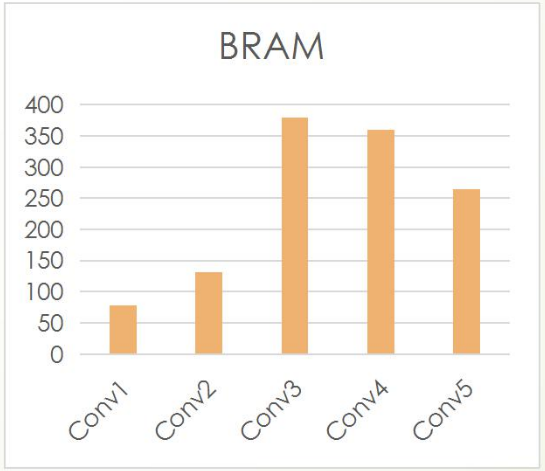
Throughput = Total ternary compute*2*Frequency

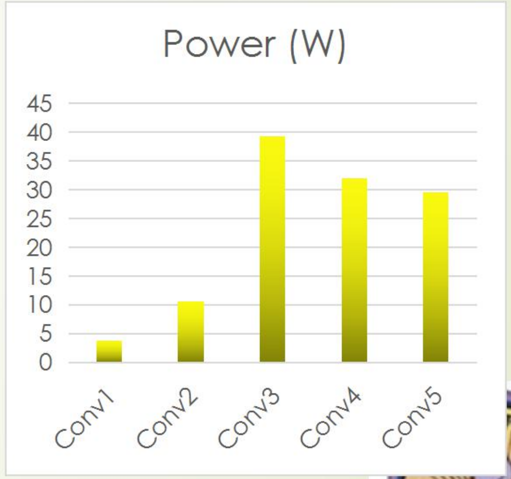
Results

Layer	LUTS	LUTS Util %	FFs	FF Util %	BRAM	BRAM Util%	DELAY (ns)	Freq(MHz)	Power (W)
Conv1	36014	5.0	34012	2.3	78	4.1	1.83	543.7	3.7
Conv2	52217	7.0	53509	3.7	132	7.0	2.2	453.3	10.4
Conv3	204038	28.6	198968	13.9	380	20.2	2.2	453.3	39.2
Conv4	142668	20.0	139737	9.8	360	19.1	2.2	453.3	31.9
Conv5	142368	19.9	141542	9.9	264	14.0	2.2	453.3	29.5
Total	577305	81.0	567768	39.8	1214	64.5	2.2	453.3	









Conclusion

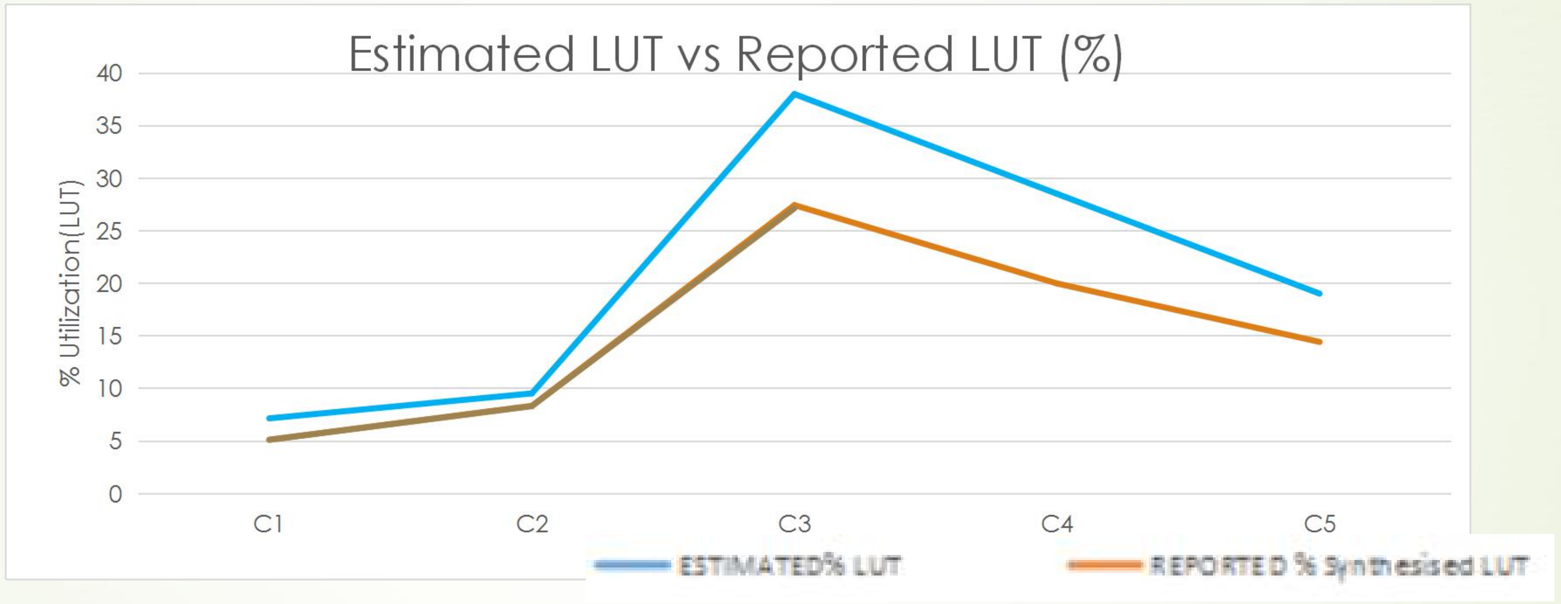
- CNNs are highly compute intensive.
- Speed and Accuracy are key factors ADAS systems.
- Ternarization helps in improving the throughoutput with almost same Accuracy.
- A Scalable Architecture for CNNs has been implemented.
- There are high fanout issues in implementation which we are trying to fix it.

References

- [1] FIST: A Framework to Interleave Spiking neural networks on CGRAs Tuan Ngyen, Syed M. A. H. Jafri, Masoud Daneshtalab, Ahmed Hemani, Sergei Dytckov, Juha Plosila, and Hannu Tenhune, Turku Centre for Computer Science, University of Turku, Finland, Royal Institute of Technology, Sweden, 2015 23rd Euromicro International Conference on Parallel, Distributed, and Network-Based Processing
- [2] DianNao: A Small-Footprint High-Throughput Accelerator for Ubiquitous Machine-Learning Tianshi Chen, Zidong Du, Ninghui Sun, Jia Wang, Chengyong Wu, Yunji Chen SKLCA, ICT, China, Olivier Temam Inria, France, ASPLOS '14, March 1–5, 2014, Salt Lake City, Utah, USA.
- [3] A CGRA-based Approach for Accelerating Convolutional Neural Networks Masakazu Tanomoto, Shinya Takamaeda-Yamazaki, Jun Yao, and Yasuhiko Nakashima, Graduate School of Information Science, Nara Institute of Science and Technology, Nara, Japan, 2015 IEEE 9th International Symposium on Embedded Multicore/Many-core Systems-on-Chip.
- [4] Li, Huimin, et al. "A high performance FPGA-based accelerator for large-scale convolutional neural networks." *Field Programmable Logic and Applications (FPL), 2016 26th International Conference on*. IEEE, 2016
- [5] H. Alemdar, N. Caldwell, V. Leroy, A. Prost-Boucle, and F. P´etrot. Ternary Neural Networks for Resource-Efficient AI Applications. CoRR, abs/1609.00222, 2016.

Back up

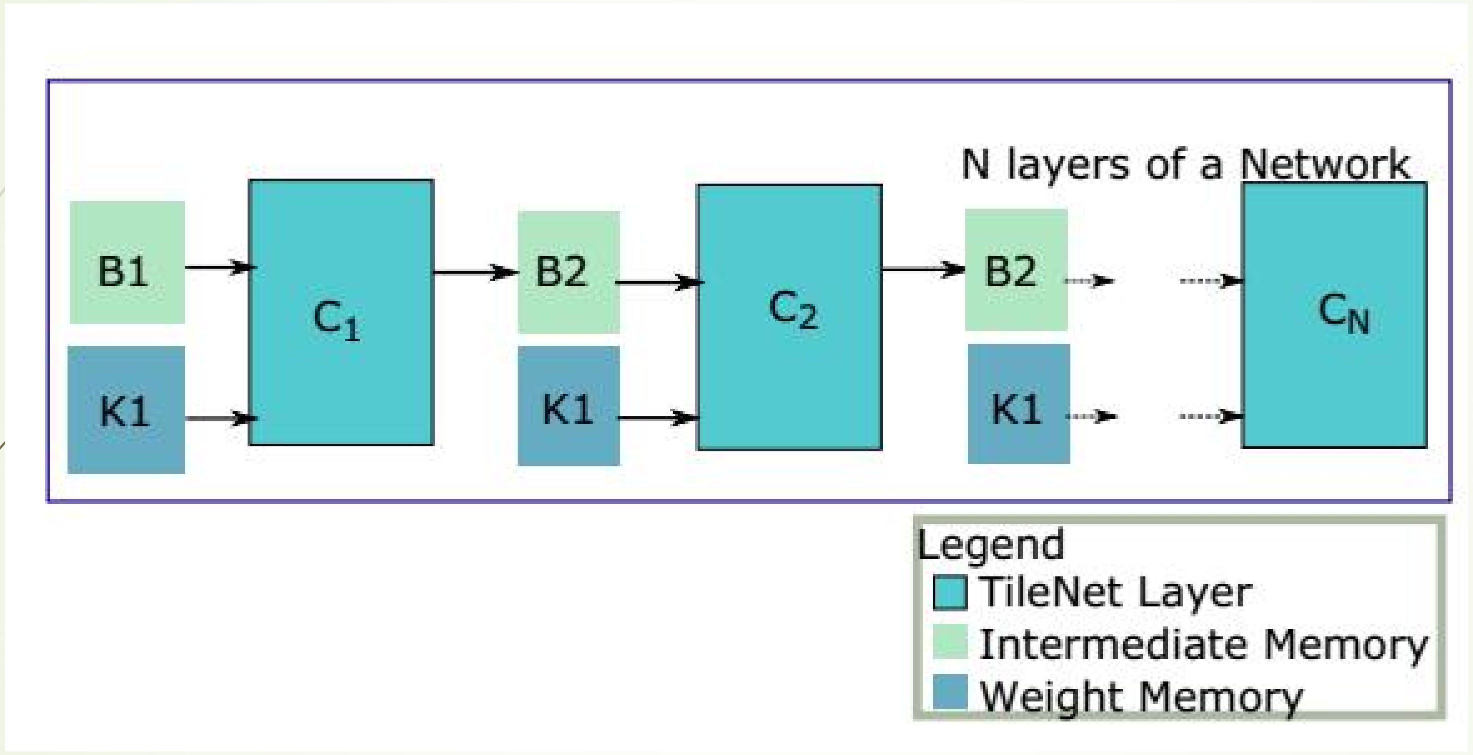
Estimation vs Implemention



CNN

- CNN is a multi-layered neural network with multiple hidden layers.
- **Input layer** is the input interface for training and test data.
- **Output layer** indicates the probability that the input data belong each classification category
- **Hidden layers** consist of **convolution layers, sub-sampling layers** (pooling and max-out layers), and full-connected layers.
- In **convolution layer**, the value of each point is obtained by a convolution of a corresponding sub-region in the previous layer and a weight vector.
- **Pooling layer** is a condensed vector of representative values in the previous layer, which is generated by sampling of a maximum value or an average value in every small region in the previous layer
- In **full-connected layers**, all points in a layer are connected to all points in the adjacent layer, in order to determine the values of the output layer by using the feature map information in previous convolution and sub-sampling layers.

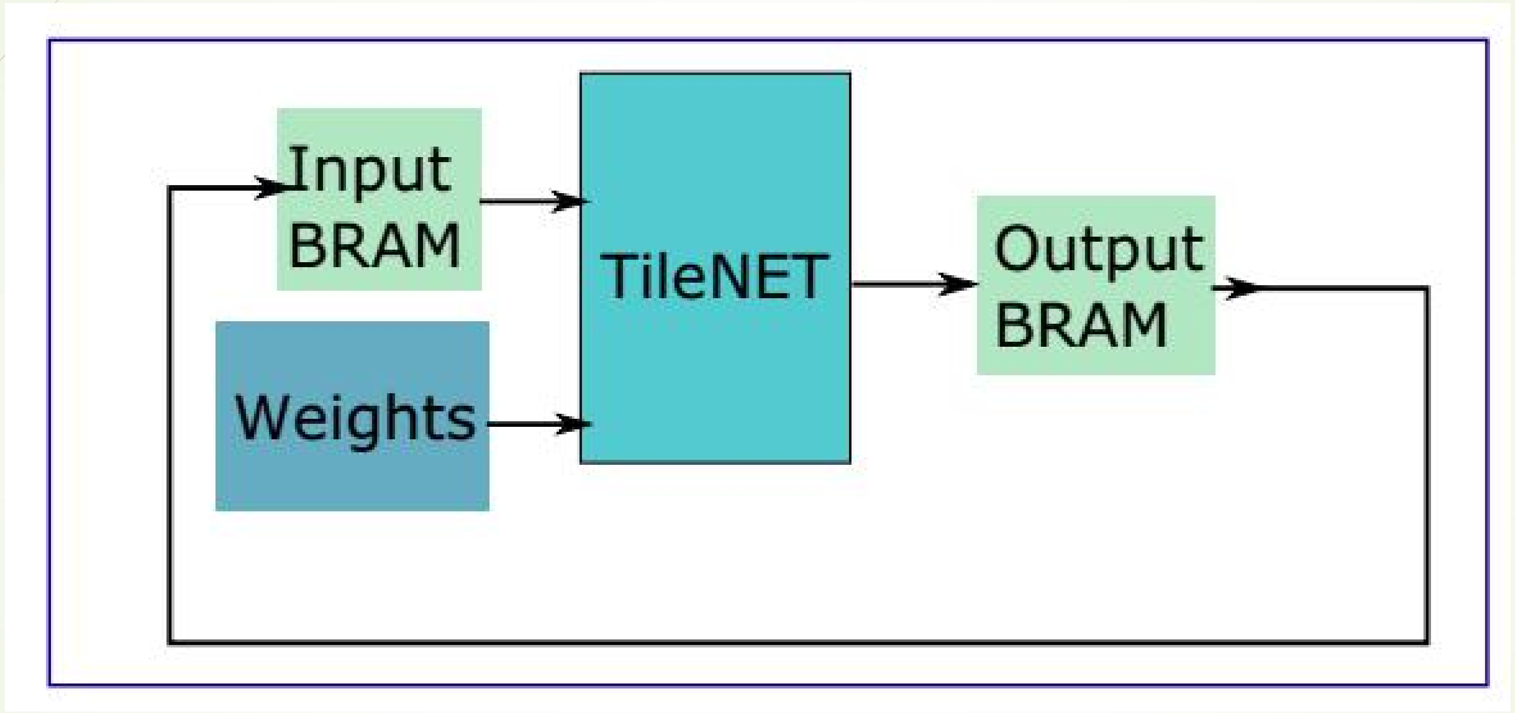
Streamlined Architecture



Streamlined Architecture

- In the Streaming mode, all the layers of a given network model co-exist on the device, as shown in Figure.
- In each layers, tiles sufficient enough to supply data to trigger computations in the following layers.
- Consequently, all the layers are run in parallel with limited amount of parallelization provided to each layer. ,
- The partial output feature maps from the each layers are stored in On-Chip memories and passed to next layer.

Systolic Architecture



Systolic Architecture

Systolic Architecture

- In the systolic mode, all the resources on the device are used to implement a convolution layer with the maximum number of tiles as can be accommodated on the device, as shown in Figure.
- The execution of the network model is therefore serialized, with one layer being completed entirely before initiating computation on the next layer in a systolic fashion.
- As a result, layers are executed one after the other with the output of each layer stored onto on-chip memory. The contents of this memory acts as input to the next layer.

Estimation(Memory)

Layer	N	M	K	Input Memory(BRAMS)	Kernel Memory(BRAMS)	Total	% Utilization
Conv1	3	96	11x11	6	72	78	4.14893617
Conv2	48	256	5x5	36	96	132	7.021276596
Conv3	256	384	3x3	96	284	380	20.21276596
Conv4	192	384	3x3	72	288	360	19.14893617
Conv5	192	256	3x3	72	192	264	14.04255319
						1214	**64.57446809**

Estimation(Compute)

Device	xc7vx1140(Vertex 7)
LUTS	712000
FFS	1424000
BRAMS	1880

Parameters:

N	- input feature Maps
M	- output feature Maps(No. of kernel Weights)
k	- kernel width
Nt	- No. of tiles in Kernel
Rin,Cin	- input feature map width and height
Ro,Co	- output feature map width and height
C	- Parallelization factor
M/c	- No. of times layer has to run

Layer	N	M	K	Nt	Rin	Cin	Ro	Co	M/C	C value	Max Pooling	Ternary Compute	LUTS
Conv1	3	96	11x11	16	224	224	55	55	16	6	3x3	2592	50688
Conv2	48	256	5x5	4	55	55	27	27	128	2	3X3	3456	67584
Conv3	256	384	3x3	1	13	13	13	13	64	6	NA	13824	270336
Conv4	192	384	3x3	1	13	13	13	13	64	6	NA	10368	202752
Conv5	192	256	3x3	1	13	13	13	13	64	4	3X3	6912	135168
Total												**37152**	**726528**

%LUTS	102.04
Freq	500MHz
Throughput	37.15 TOPs

Throughput = Total ternary compute*2*Frequency

Tiling

- A single tile accommodates the entire compute associated with a input kernel size.
- The tile size (T) of 3x3 was considered, as for most of the layers a weight vector of size is 3x3 and it offers the optimal compute-memory ratio.
- Weight vector sizes higher than 3x3 can be implemented by partitioning it into multiples of of 3x3 Tiles.