

Convolutional Neural Networks on embedded automotive platforms: a qualitative comparison

Gianluca Brilli, Paolo Burgio, Marko Bertogna
University of Modena and Reggio Emilia

88740@studenti.unimore.it



Slide 1

GB1

Gianluca Brilli, 2018-09-13



Neural networks... for tomorrow



› Extensively adopted in the embedded world

- › Computer vision and image processing tasks,
 - object categorization and labeling



› Autonomous driving, industry 4.0

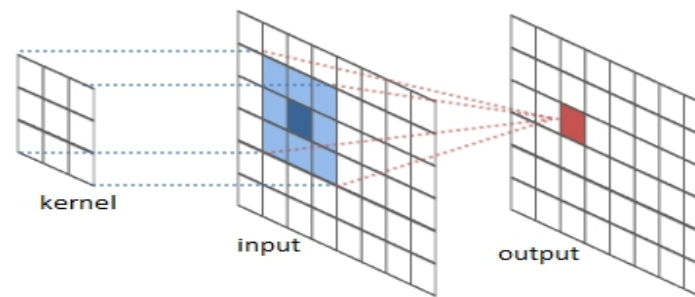




Convolution Neural Networks

$$o_{i,j}^k = \sum_{c=0}^{D_{in}} \sum_{h=0}^{K_H} \sum_{w=0}^{K_W} (w_{h,w,c}^k x_{i+h,j+w,c}) + b_k$$

- › High computational cost
 - › Simple "sum-of-products" structure
 - › No inter-pixel dependencies

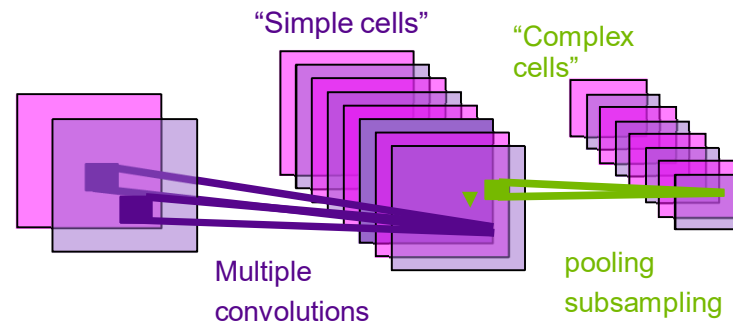
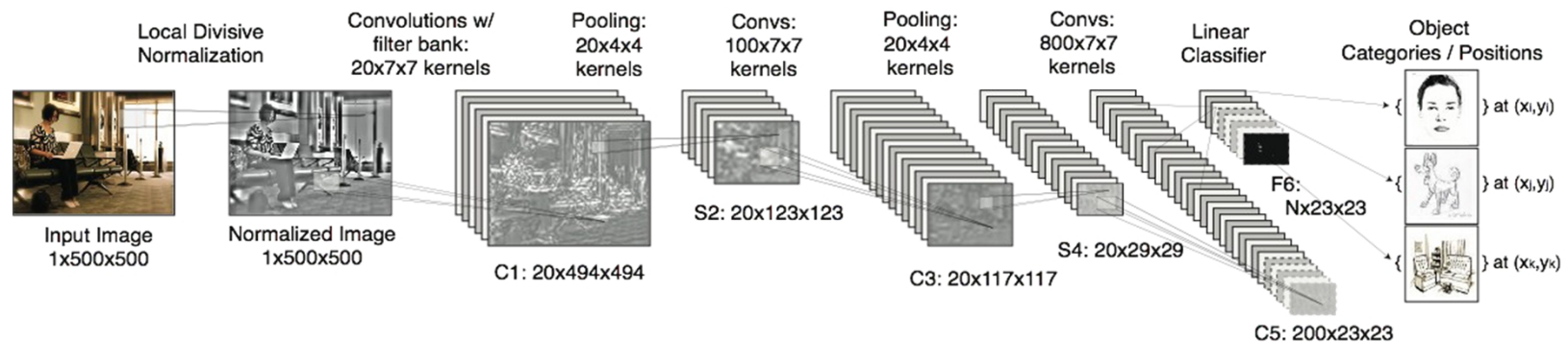


Highly Massively parallelizable!!



A lot of stuff to do...

- › Multiple bidimensional layers
- › Huge number of multiply-accumulate (MAC) operation
 - on thousands of pixel of an input image
- › CNN: convolutional neural networks





..at low SWaP



Training is "easy"

powerful servers

"big data"-sets

Inference is an **issue**

on-vehicle ECUs, in-plant boards
constrained in Size, Weight and Power





This (ongoing) work

Profile open-source packages...

...of state-of-the-art (C)NNs...

...on automotive platforms

Three categories

- › Present (Embedded GP-GPUs)
- › (Next) future (Reconfigurable/FPGAs)
- › (Next-next) future



State-of-the-art embedded platforms



Graphics Processing Units



Programmable logics

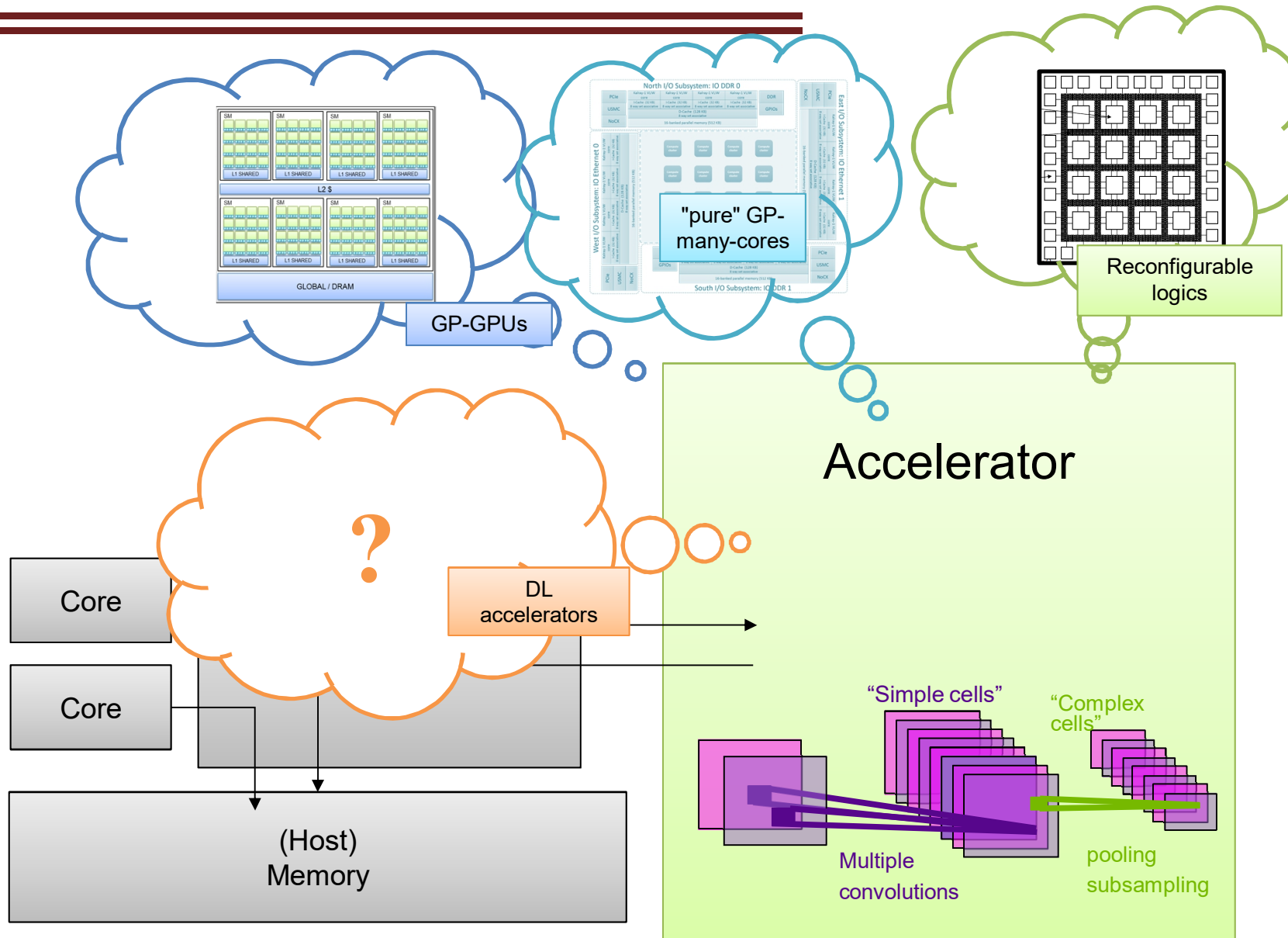


"Pure" Many-Cores

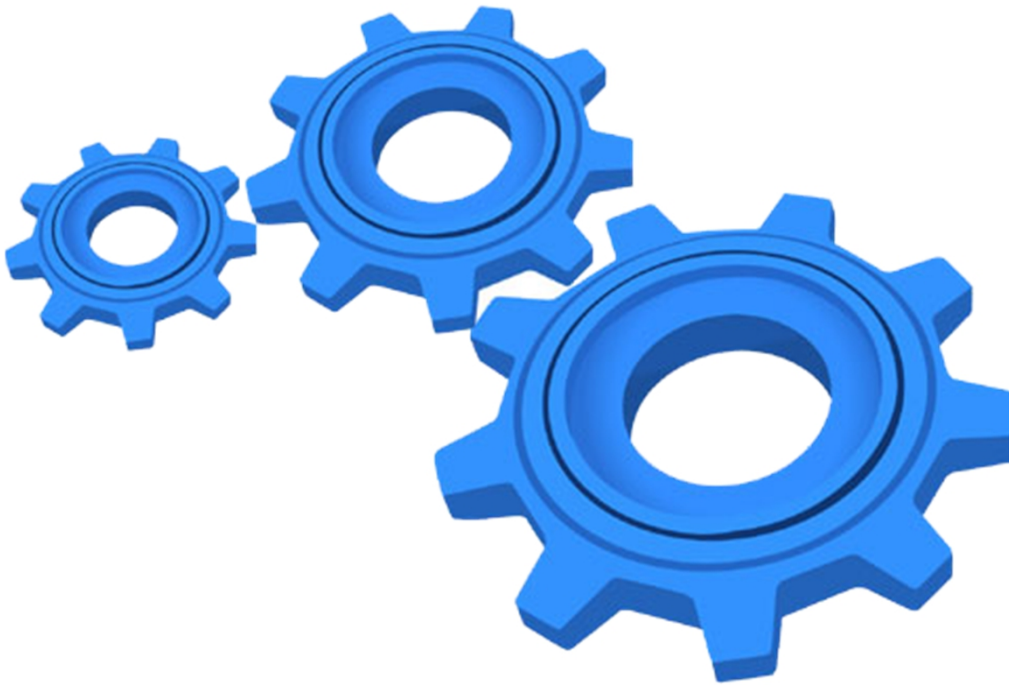




...which architecture?



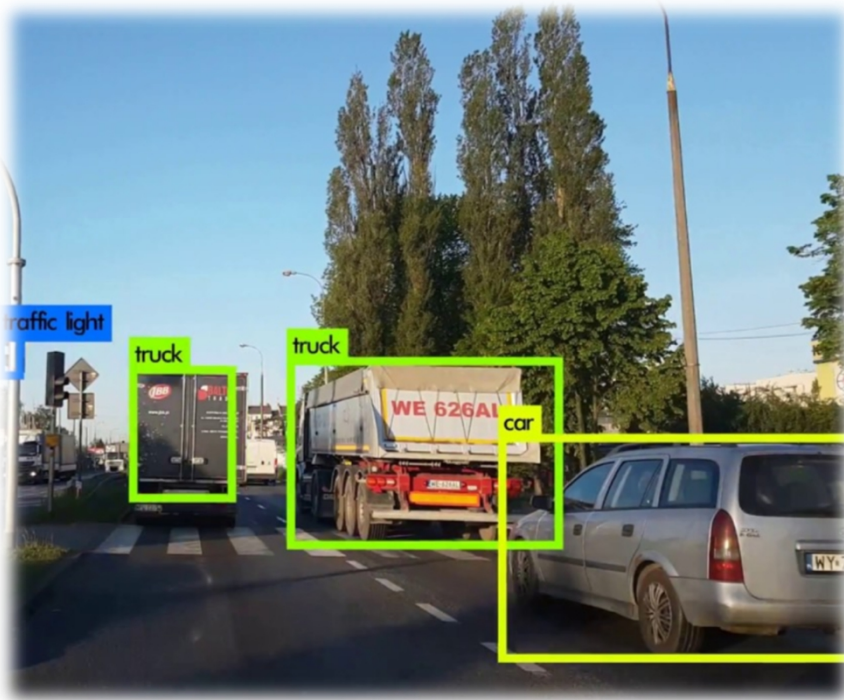
Setup





CNN for Object Detection

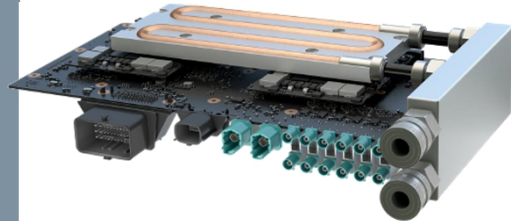
J. Redmon's YOLO: Real-Time Object Detection



- › **YOLO**: *full model*, 23 conv layers
- › **Tiny-YOLO**: *reduced model*, 9 conv layers



Target platforms

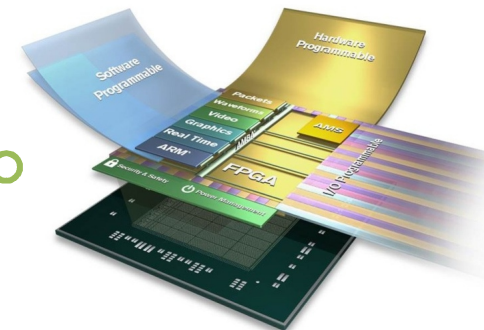


Tegra X2/Parker SoC

- › Drive PX2 for autonomous driving
- › 4 x ARM Cortex A57 + 2 x Denver
- › Pascal GPU

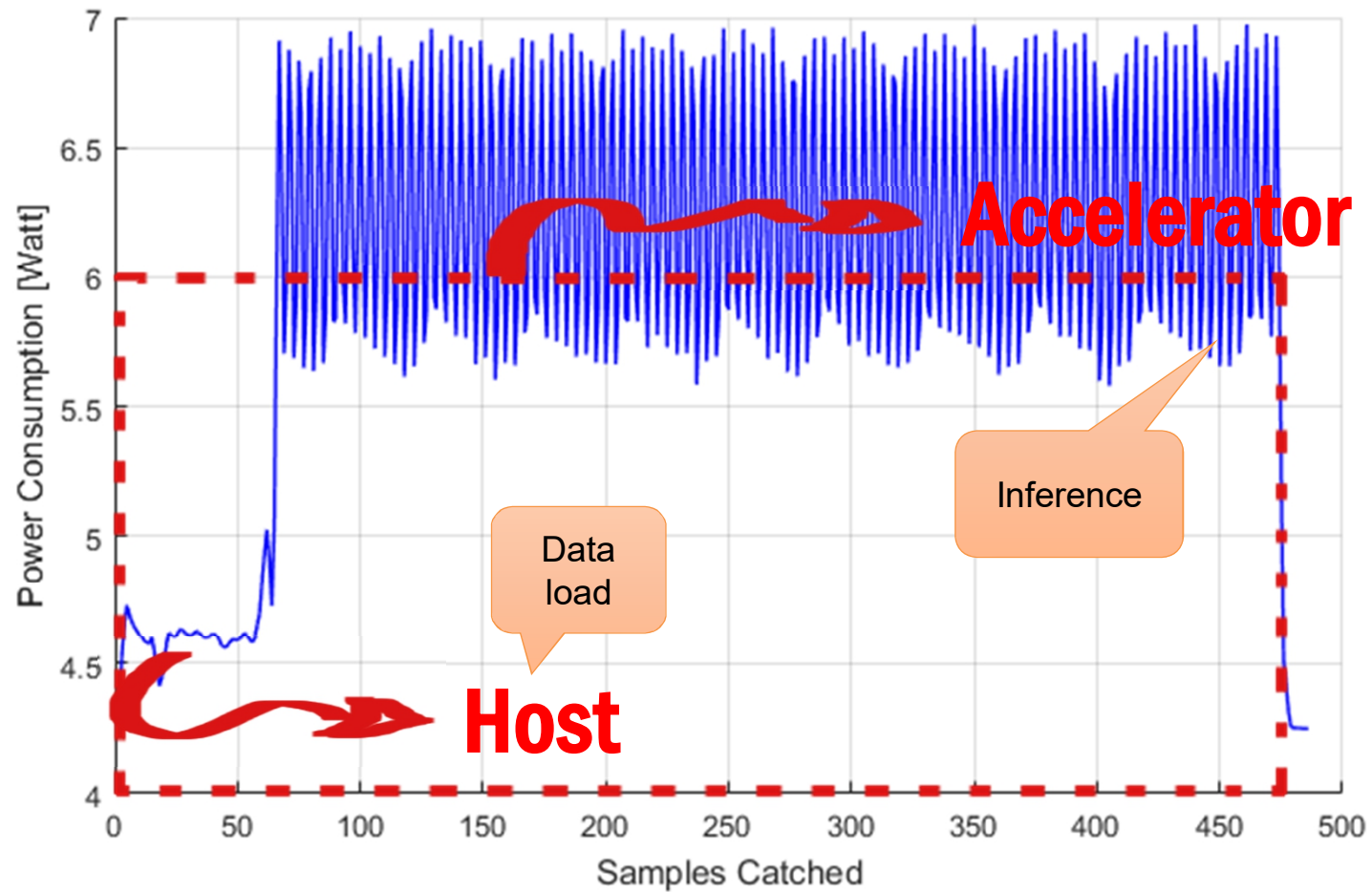
Zynq Ultrascale+

- › 4 x ARM Cortex A53 + 2 x R5
- › Mali GPU
- › FPGA fabric





Typical benchmark behavior

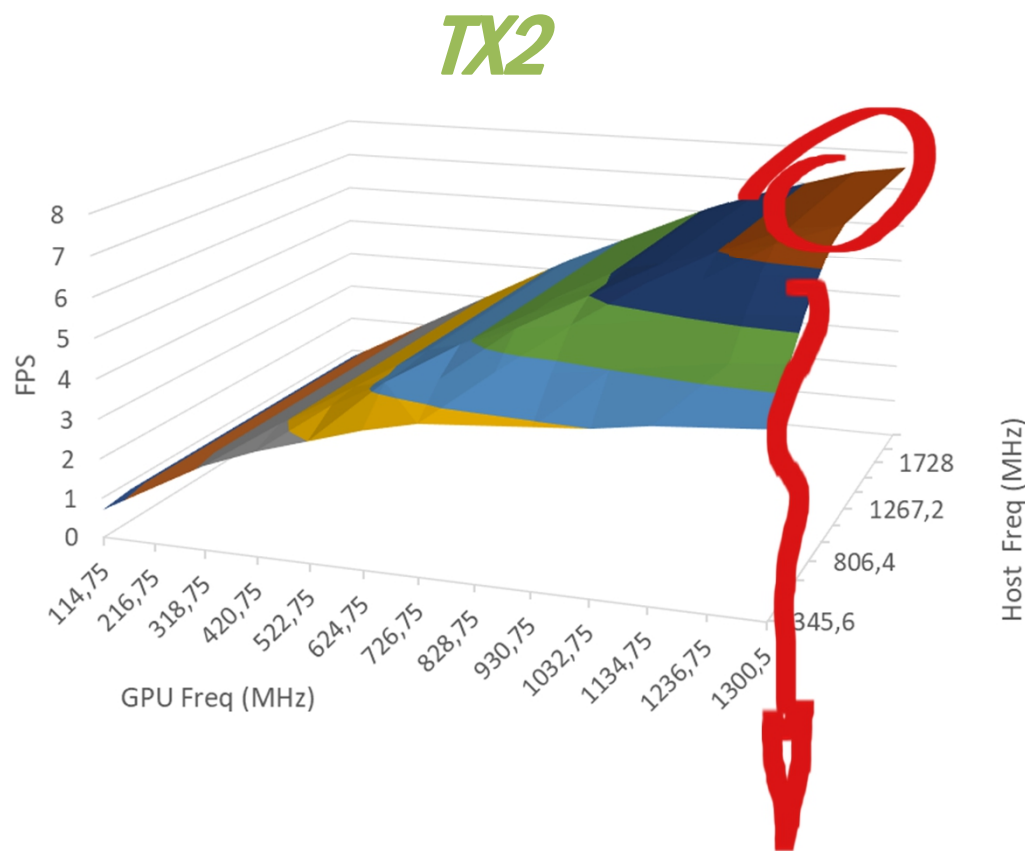


**..some numbers
(at last)**





YOLO - Frames-per-Second



XU+

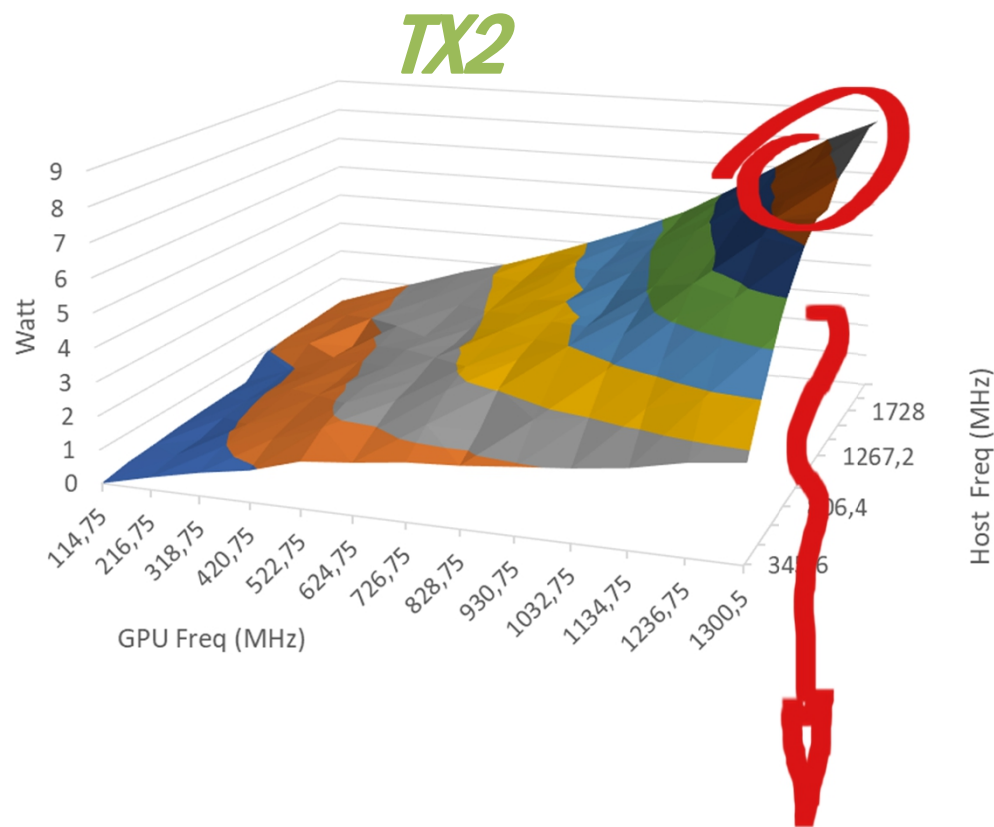
PS Freq. [MHz]	PL Freq. [MHz]	Tput [FPS]
2400	200	6,67



~ 8 FPS: GPU 1,2x faster



YOLO – Power (inf.)



XU+

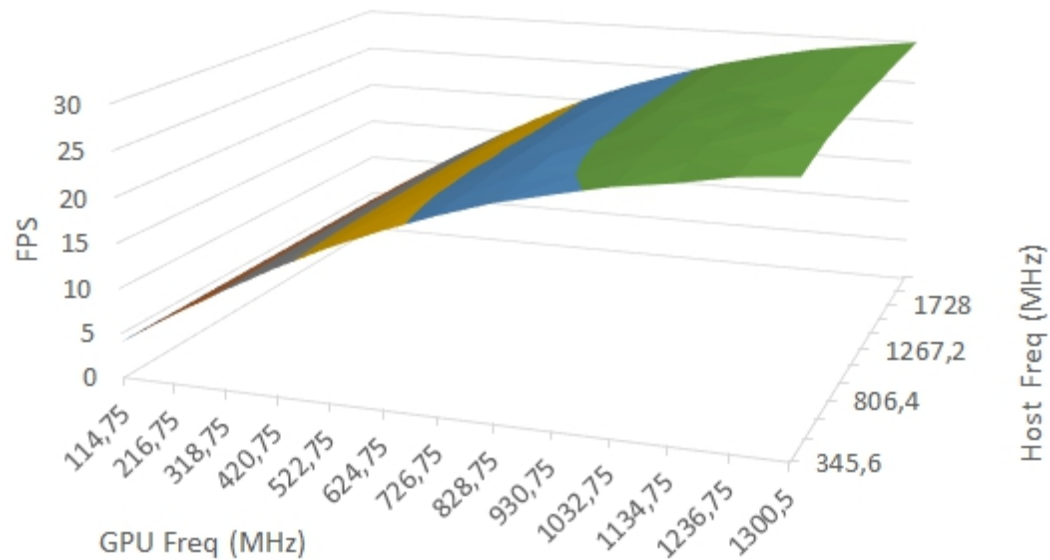
PS Freq. [MHz]	PL Freq. [MHz]	Pinf [Watt]
2400	200	0,69

~ 8 Watt: GPU 11,5x more power



Tiny-YOLO – FPS

- › 9 layers
- › Reported 57.1% mean avg Precision vs 78.6% Yolo



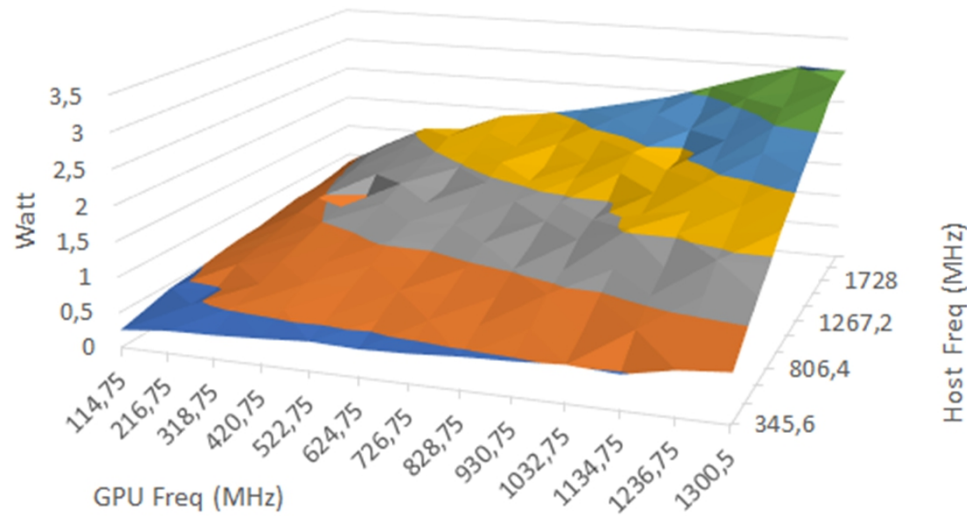
PS Freq. [MHz]	PL Freq. [MHz]	Tput [FPS]
2400	200	22,68

~ 30 FPS: 1,32x faster



Tiny-YOLO – Power (inf.)

- › 9 layers
- › Reported 57.1% mean avg Precision vs 78.6% Yolo



PS Freq. [MHz]	PL Freq. [MHz]	Pinf [Watt]
2400	200	1,15

~ 3 Watt: 2,6x more power



Discussion

- › On FPGAs, highly dependant on NN engine
- › GPU use Caffe engine, FPGA xFDNN engine
 - Highly-optimized for GPU (impressive performance for ZynqNet and AlexNet)
 - Thanks to FP data
 - 16-bit int ops on FPGA
- › Caffè engine + demos from Xilinx not programmable
- › GPU still reference for performance
- › XU+ ~~carrier board~~ SoC 1 up to order of magnitude more power efficient



Future Works



- › **Evaluate other platforms**
 - Currently: Kalray Bostan MPPA
- › **Exploiting some optimizations on FPGA**
 - For example binarization
- › **Better power measurement methodology**
 - Lauterbach?

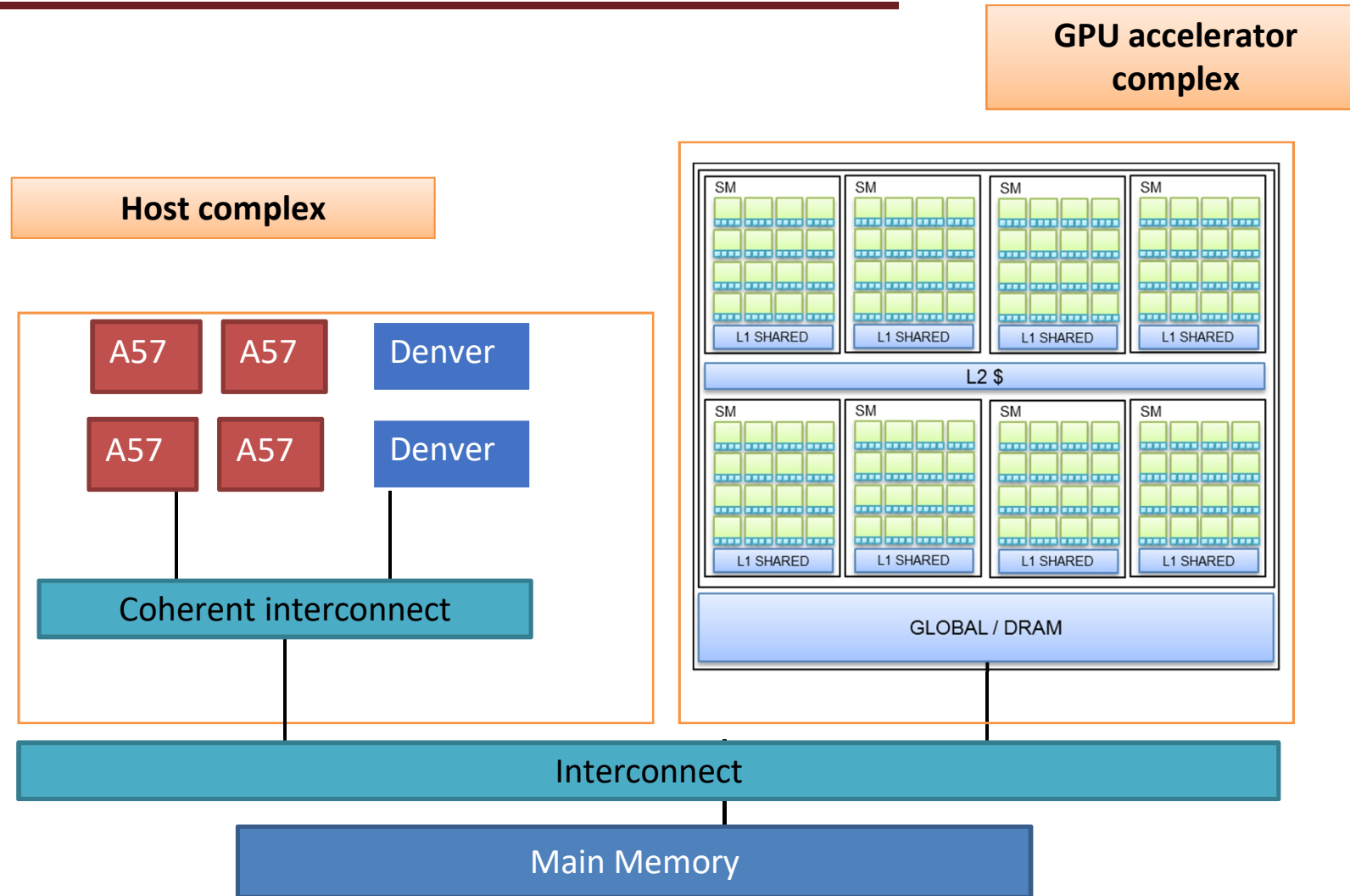


Backup





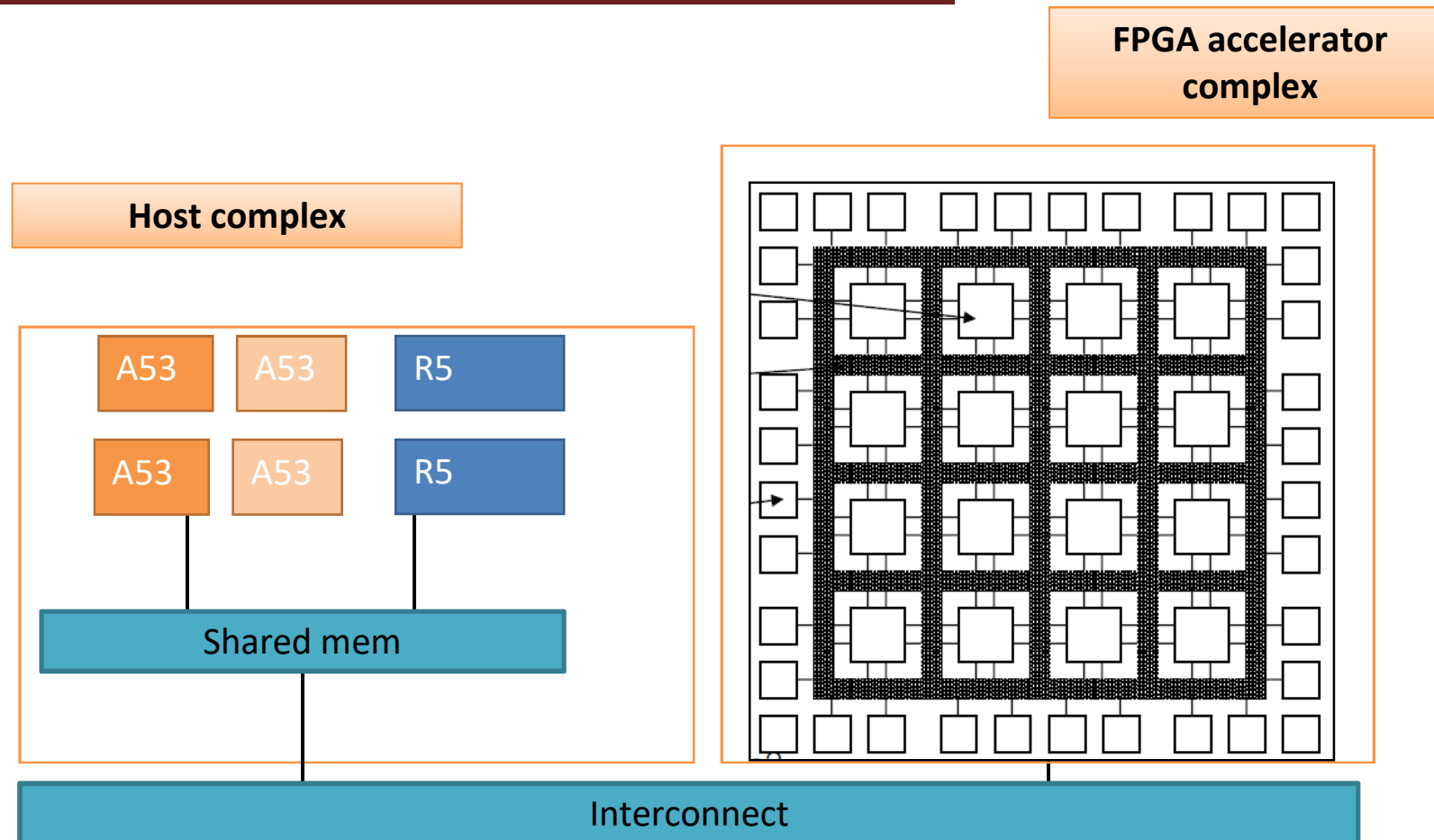
The present: NVIDIA Tegra X2



› 256 core Pascal GP-GPU



The (next) future: Xilinx Ultrascale+



- › Xilinx Ultrascale+ EG/EV
 - With GPU Mali-400MP2



CNN frameworks

› GPU:

- › J.Redmon 's **Darknet**
- › HiPeRTLlab 's **tkDNN**
- › J.Yangqing 's **Caffe**

› FPGA:

- › D.Gschwend 's **ZynqNet CNN Accel**
- › D.Wang 's **PipeCNN**
- › Xilinx inc 's **xfDNN**



Accuracy

› Object Detection: Mean Average Precision (mAP)

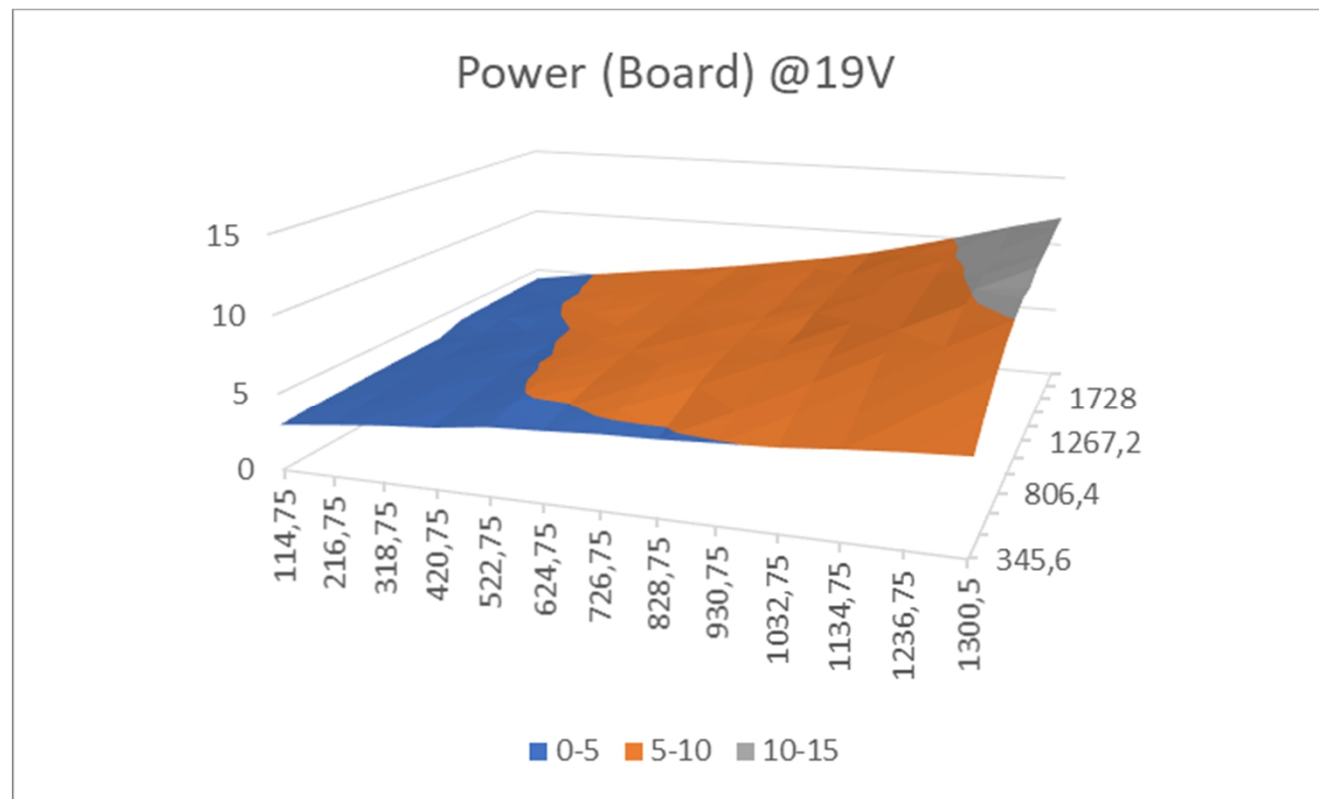
Network	mAP
YOLO	76,8
Tiny-YOLO	57,1

› Image Classification: Top-5 Accuracy

Modello	Top-5 Accuracy
AlexNet	76%
ZynqNet	83%



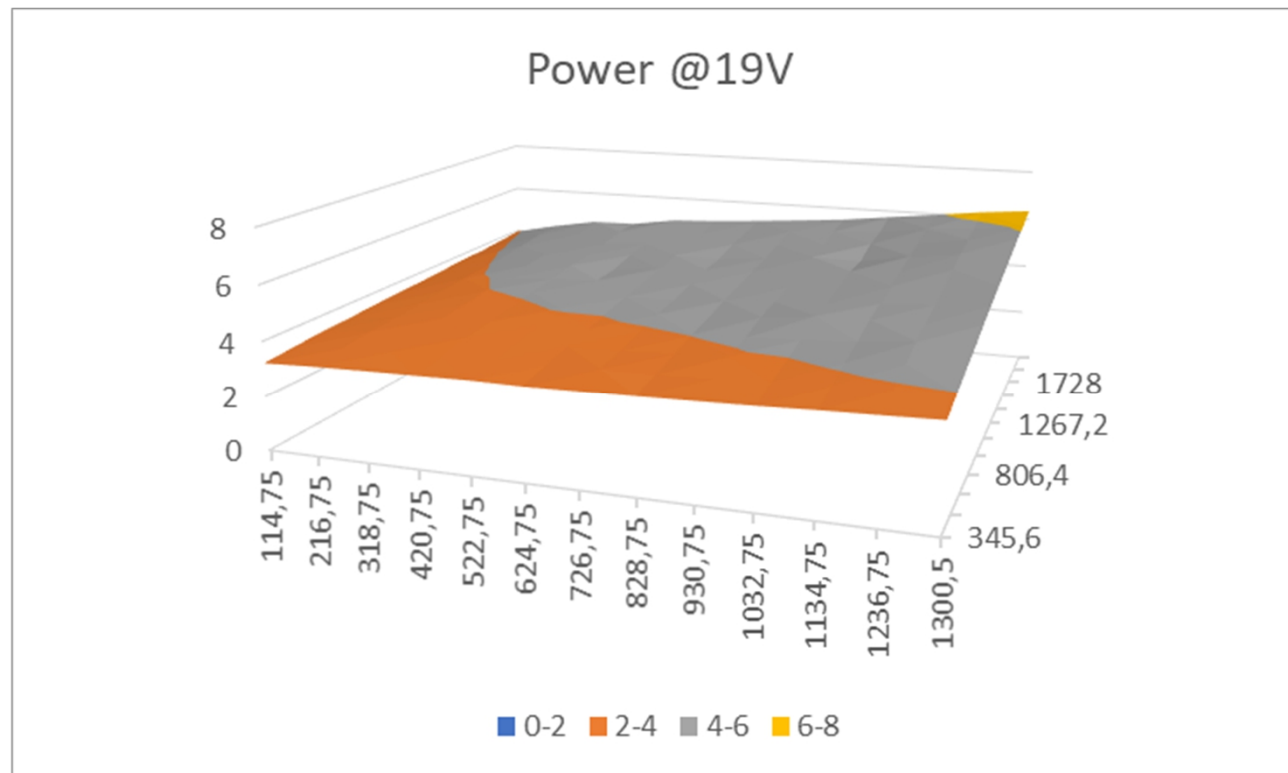
Yolo on Tegra X2: power





Yolo tiny on Tegra X2: power

- › 15 layer (half than Yolo)
- › Reported 57.1% mean avg Precision vs 78.6% Yolo





Yolos on XU+

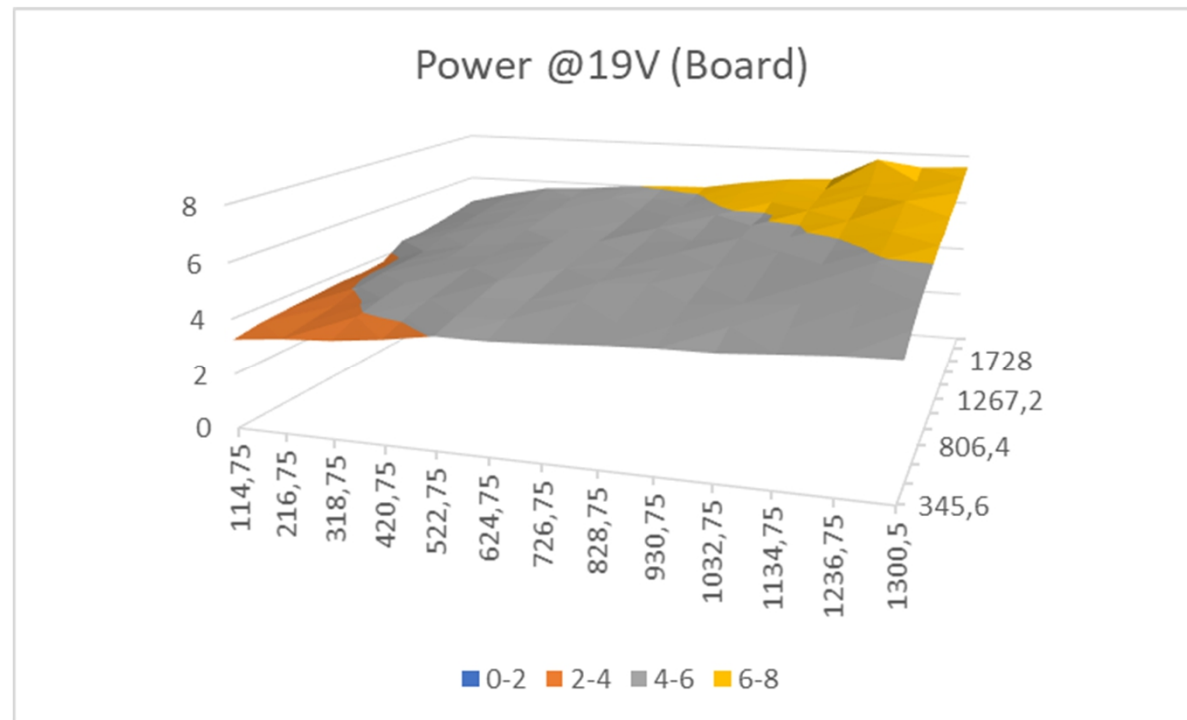
TABLE II
YOLO ON ZYNQ ULTRASCALE+

Network	PS Freq. [MHz]	PL Freq. [MHz]	T.put [FPS]	P_{board} [Watt]	P_{inf} [Watt]
YOLO	2400	200	6,6728	23,7292	0,6959
Small-YOLO	2400	200	7,9311	23,7137	0,6804
Tiny-YOLO	2400	200	22,6807	24,1798	1,1465



AlexNet on TX2

- › Classification, not detection (as Yolos)
 - Lighter
- › 5 conv. layer, fully-connected last layer, 76% precision
 - https://github.com/opencv/opencv_extra/blob/master/testdata/dnn/bvlc_alexnet.prototxt





AlexNet on XU+

› Clocked 2-3x than TX2

- No clock scaling with xfdNN engine

Avg E2E latency (ms)	
300MHz	
2.4 Ghz	16,24

Throughput (FPS)	
300MHz	
2.4 Ghz	61,554

Board average power	
300MHz	
2.4 Ghz	23,1165

SoC average power	
300MHz	
2.4 Ghz	0,5097

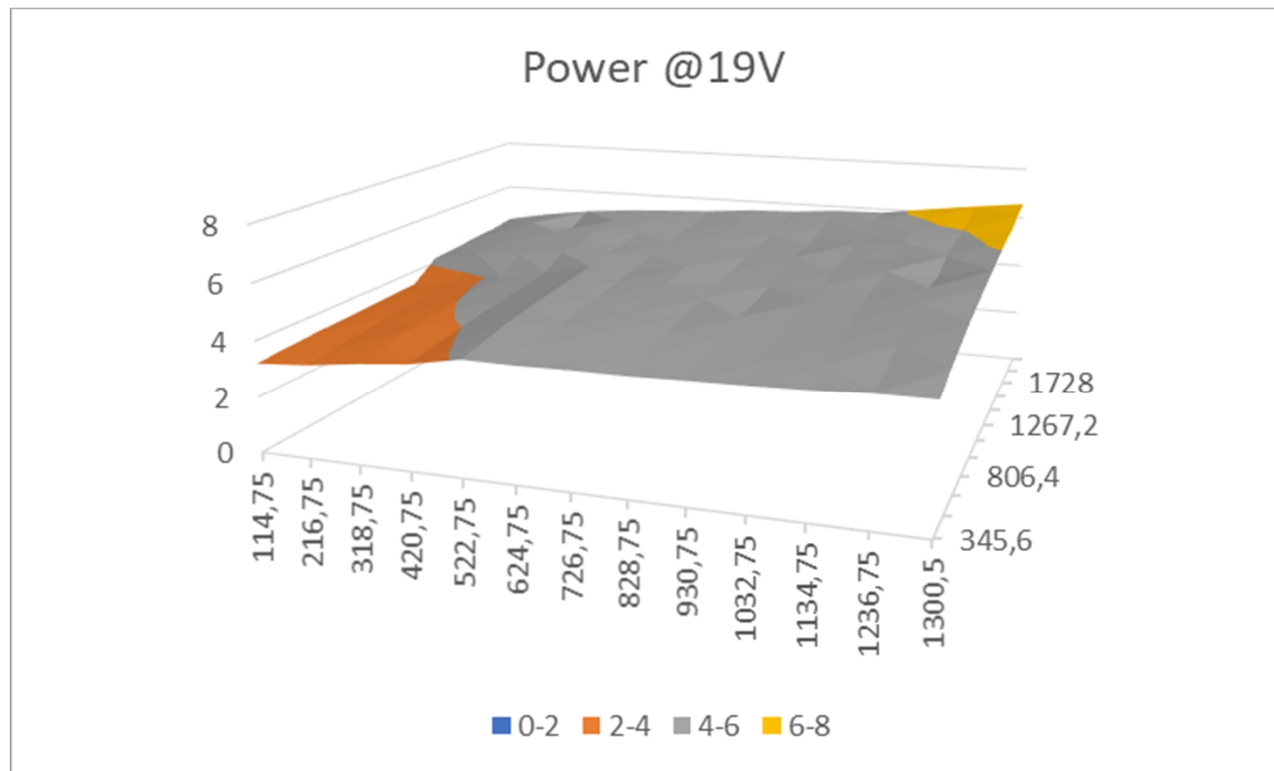
Power comparison SoC XU+ and TX2 (max freq)[Wat

	XU+		TX2	
	300MHz		114,75MHz	
2.4 Ghz	0,5097	1,11 Ghz		0,68751



ZynqNet on Tegra X2

- › Classification
- › 28 layers, 83% precision
 - <https://dgschwend.github.io/netscope/#/preset/zynqnet>





ZynqNet on UC+

- › 78% accuracy (83% on TX2)

Throughput (FPS)	
300 Mhz	
2.4 Ghz	77,47

Board average power	
300 Mhz	
2.4 Ghz	22,9953

SoC average power	
300 Mhz	
2.4 Ghz	0,3885

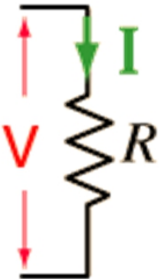


Power consumption measurement

Dynamic power: inference workload

$$P_{dyn} = \alpha C_L \times V_{DD}^2 \times f$$

Approximation


$$P = VI = \frac{V^2}{R} = I^2 R$$