Transparent Deployment of TFLite Workloads on Lightweight Many-Accelerator Architectures

Kuan-Lin Chiu, Guy Eichler, Chuan-Tung Lin

Giuseppe Di Guglielmo and Luca Carloni

June 29, 2024

COLUMBIA UNIVERSITY
IN THE CITY OF NEW YORK

CS@CU COMPUTER SCIENCE

Motivations & Goals

Optimizing Software for Accelerators in heterogeneous SoC architectures

- Modify software applications to offload specific computation kernels to accelerators
- Update the software algorithm when necessary
- Be able to run legacy software code

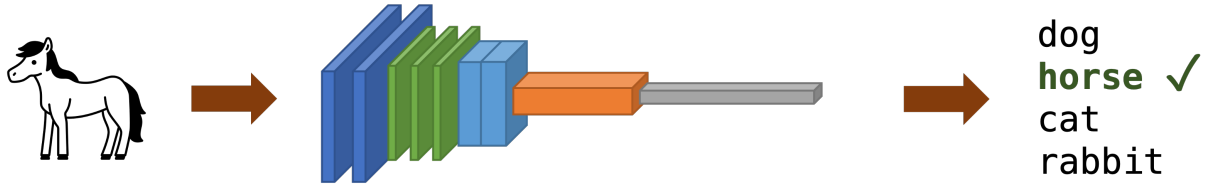
- Need: an efficient and transparent way to deploy software applications

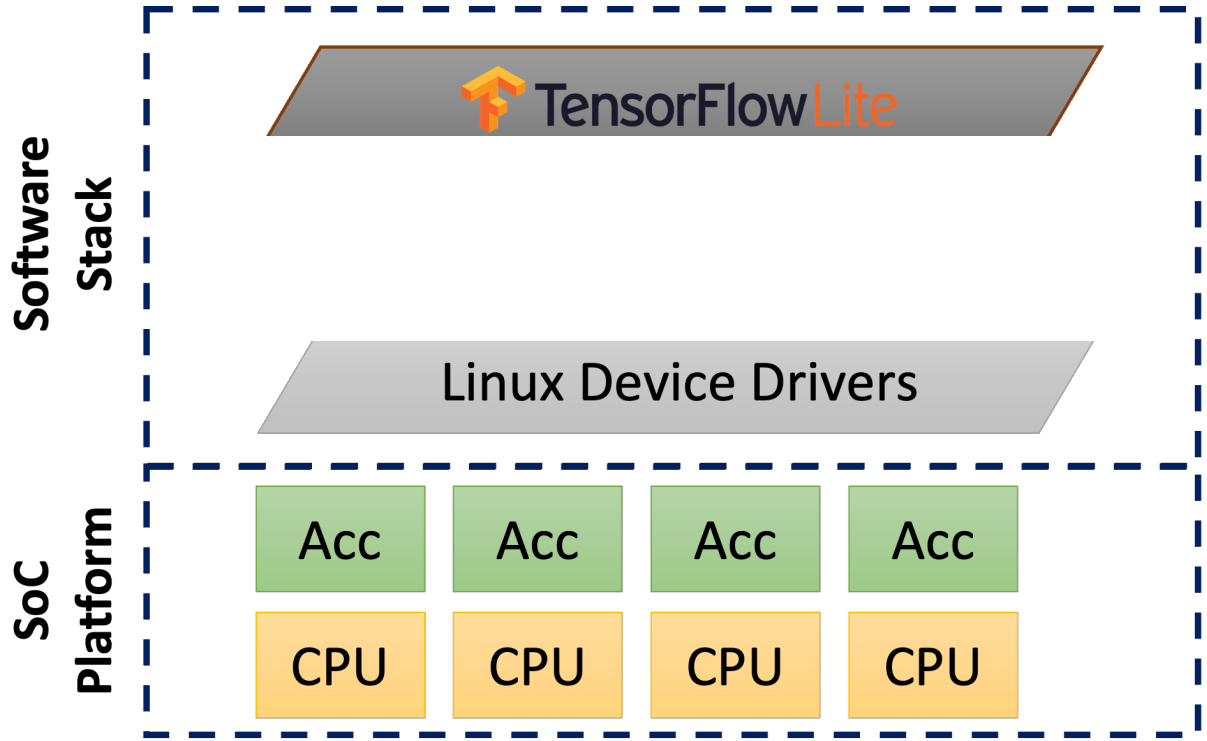
Managing ML Applications Running in Parallel on the heterogeneous SoC

- Multi-tenant scenarios in real-world applications, ex: autonomous vehicles, drones, etc
- Inevitably, applications will compete in accessing accelerators

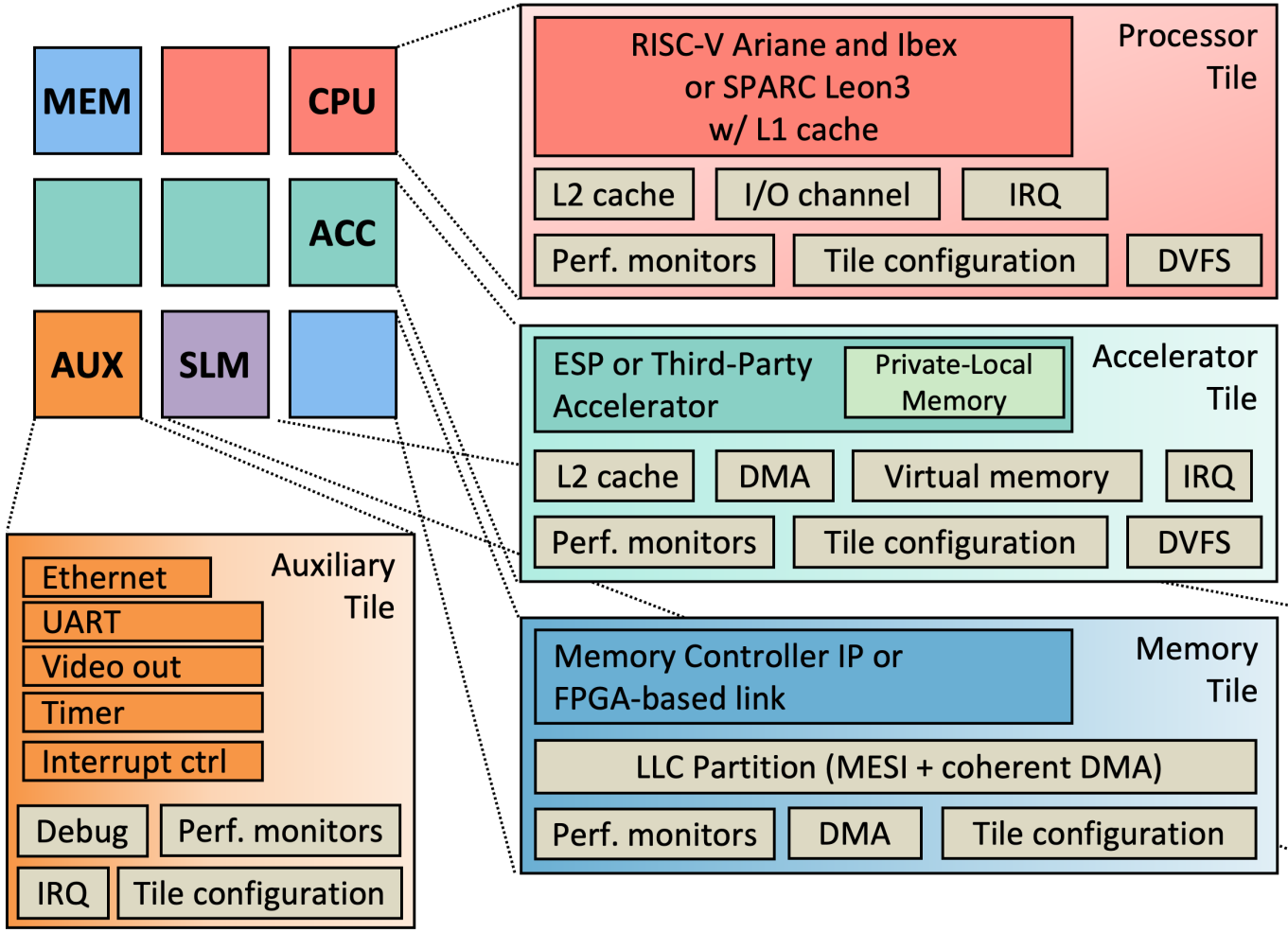
- Need: a mechanism to automatically assign available accelerators to applications

Wolt: SoC Deployment of Machine Learning Workloads



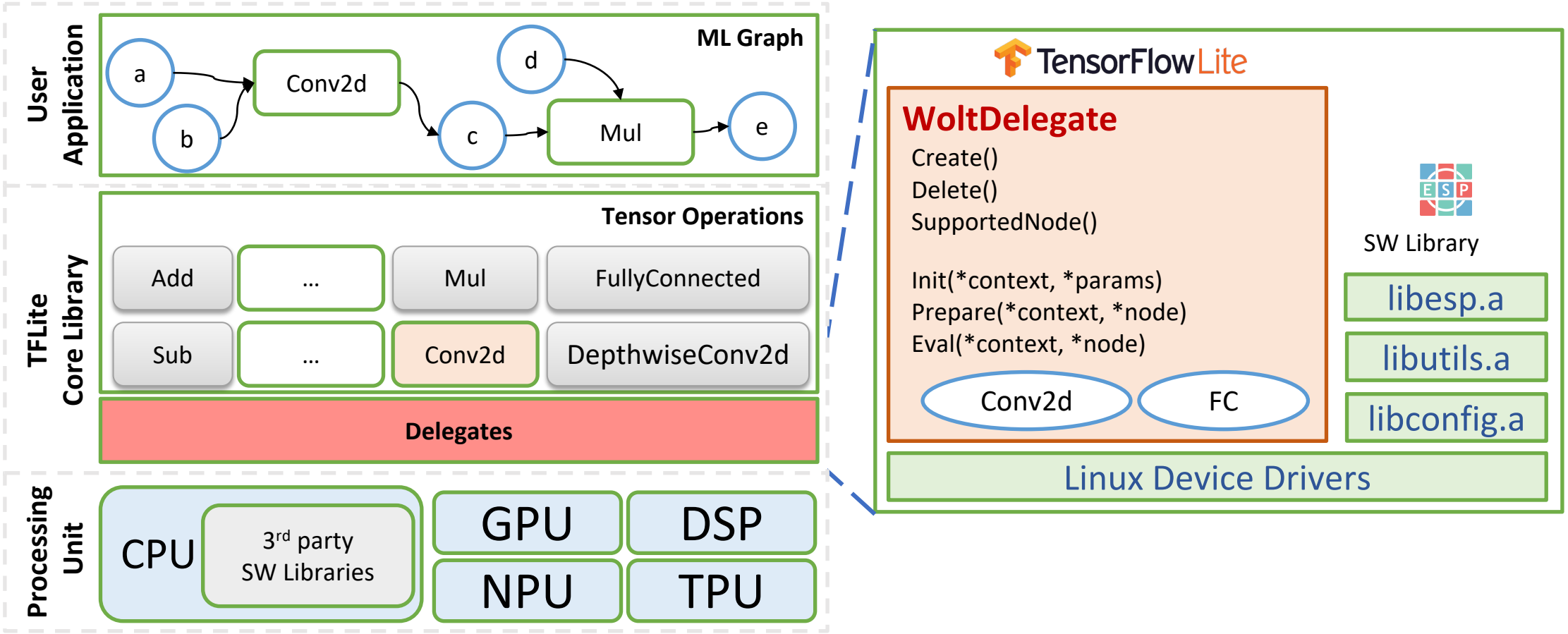


The ESP Tile-based SoC Architecture



- Scalable **NoC**
- Tile integration with **sockets**
  - Standard bus interfaces
  - Distributed **platform services**: memory access, coherence, IRQ
  - **Run-time configurable** on-chip communication

What is a Delegate? What is Wolt Delegate?

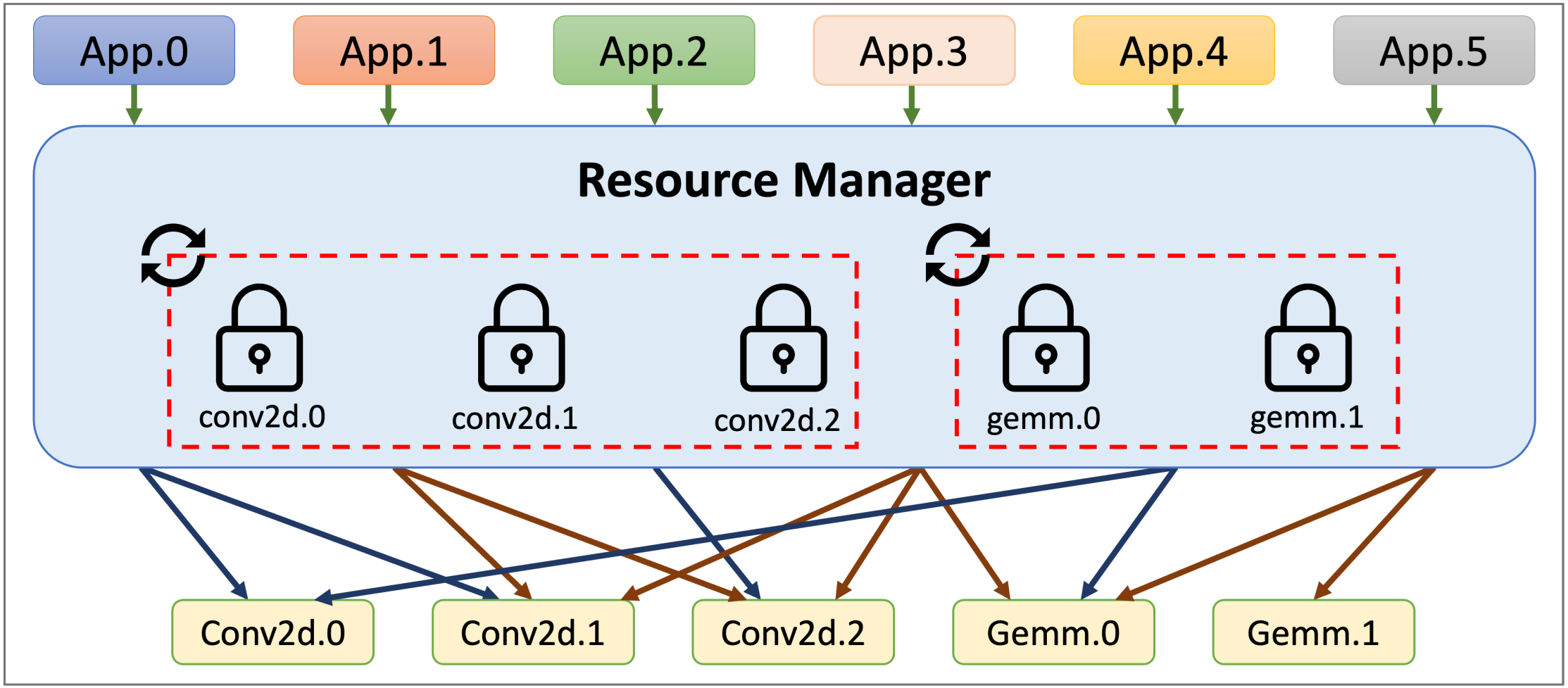


Init()
- Store indices of all delegated operations
- Initialize memories
- Initialize hardware buffer

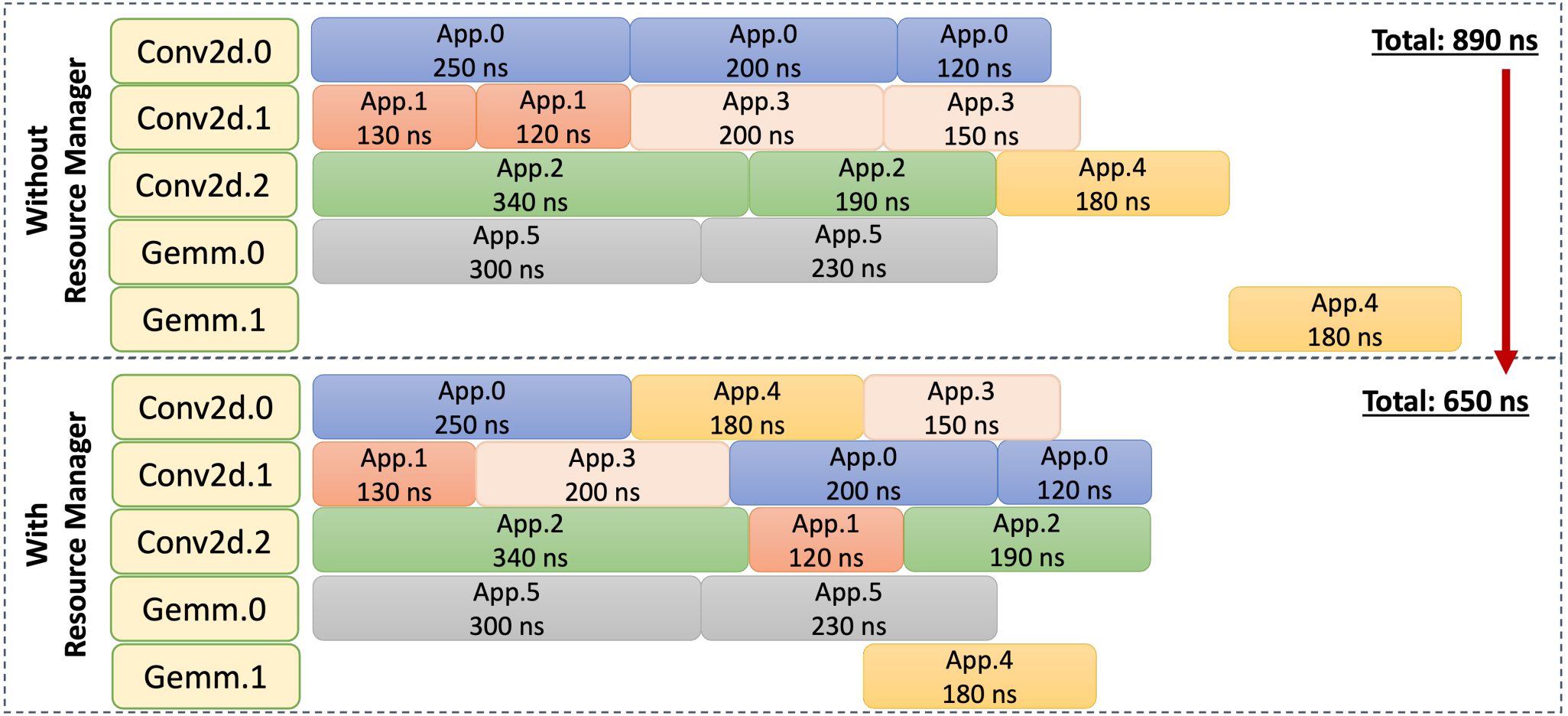
Prepare()
- Set parameters for each node (ex: dimensions of feature map and filter)

Eval()
- Execute delegated graph
- Invoke accelerators

The Wolt Resource Manager

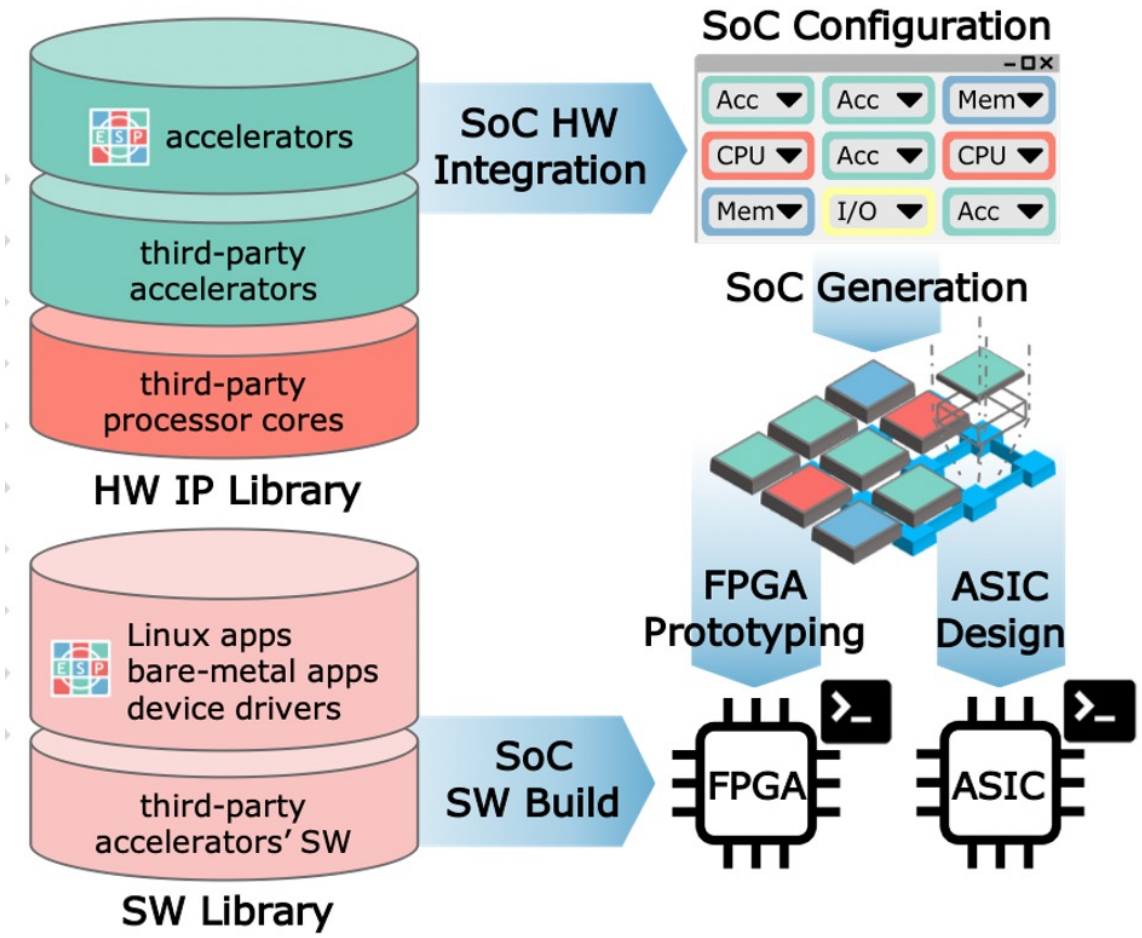


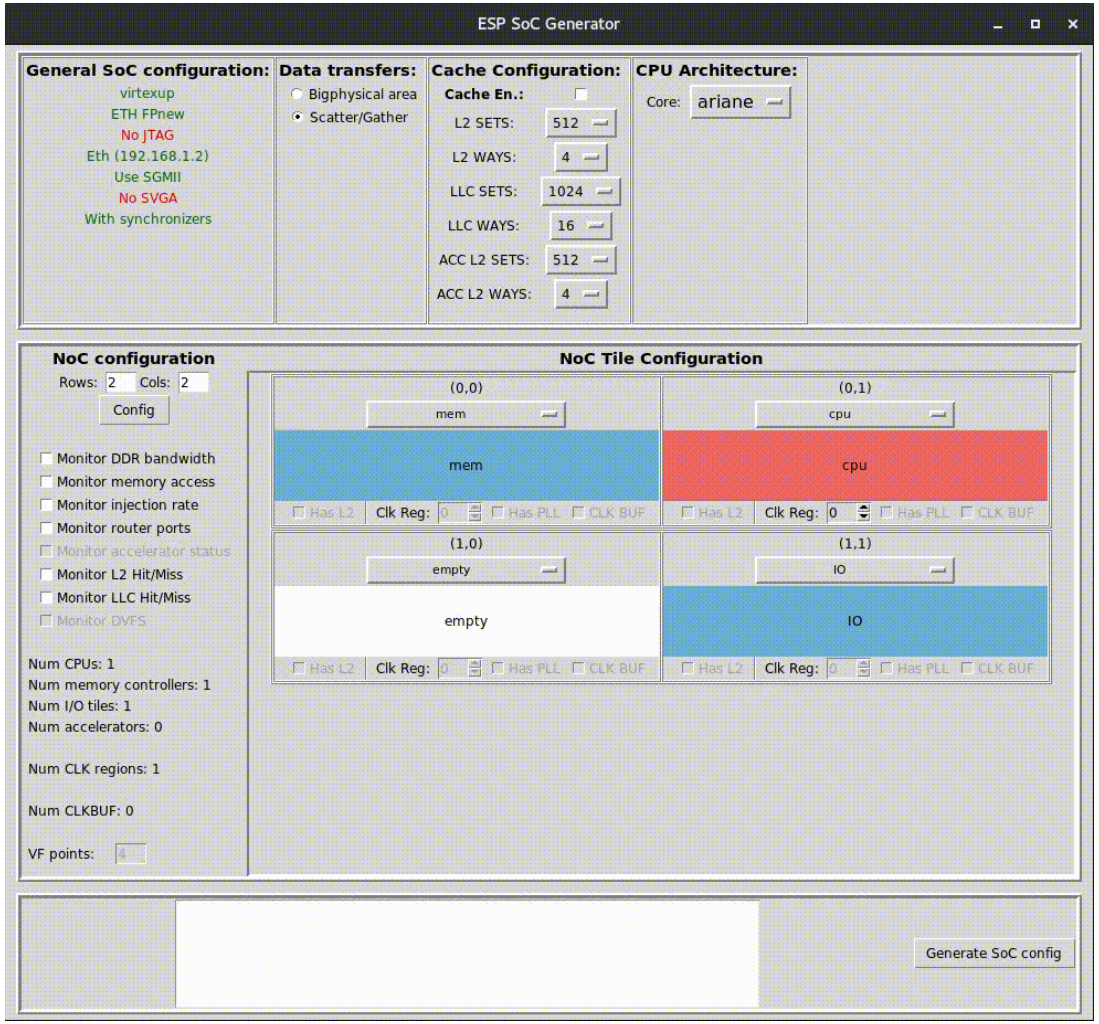
The Wolt Resource Manager: Example



- A better accelerator assignment improves performance

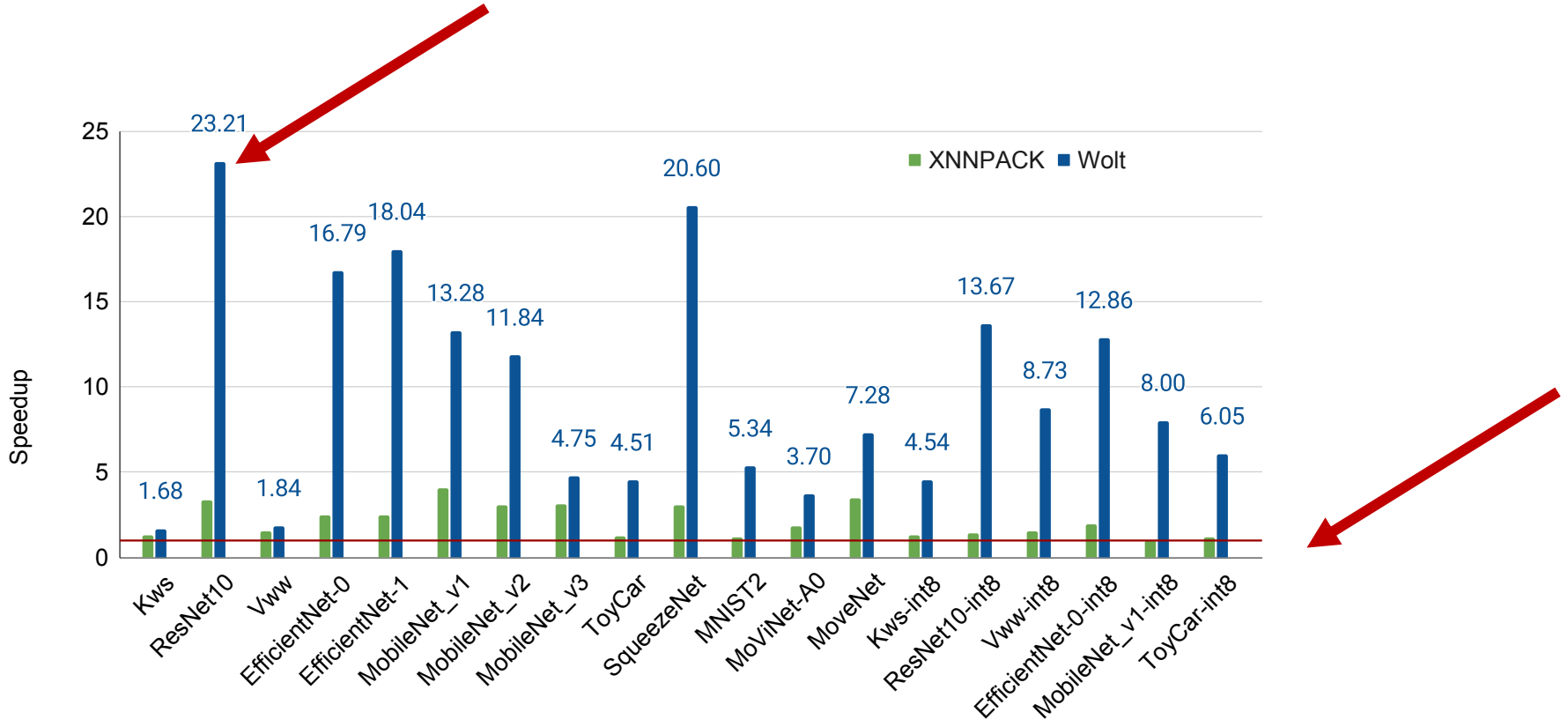
ESP SoC Flow


accelerators
third-party accelerators
third-party processor cores
HW IP Library
SoC HW Integration
SoC Configuration
Acc
Acc
Mem
CPU
Acc
CPU
Mem
I/O
Acc
SoC Generation
FPGA Prototyping
ASIC Design
Linux apps
bare-metal apps
device drivers
third-party accelerators' SW
SW Library
SoC SW Build
FPGA
ASIC

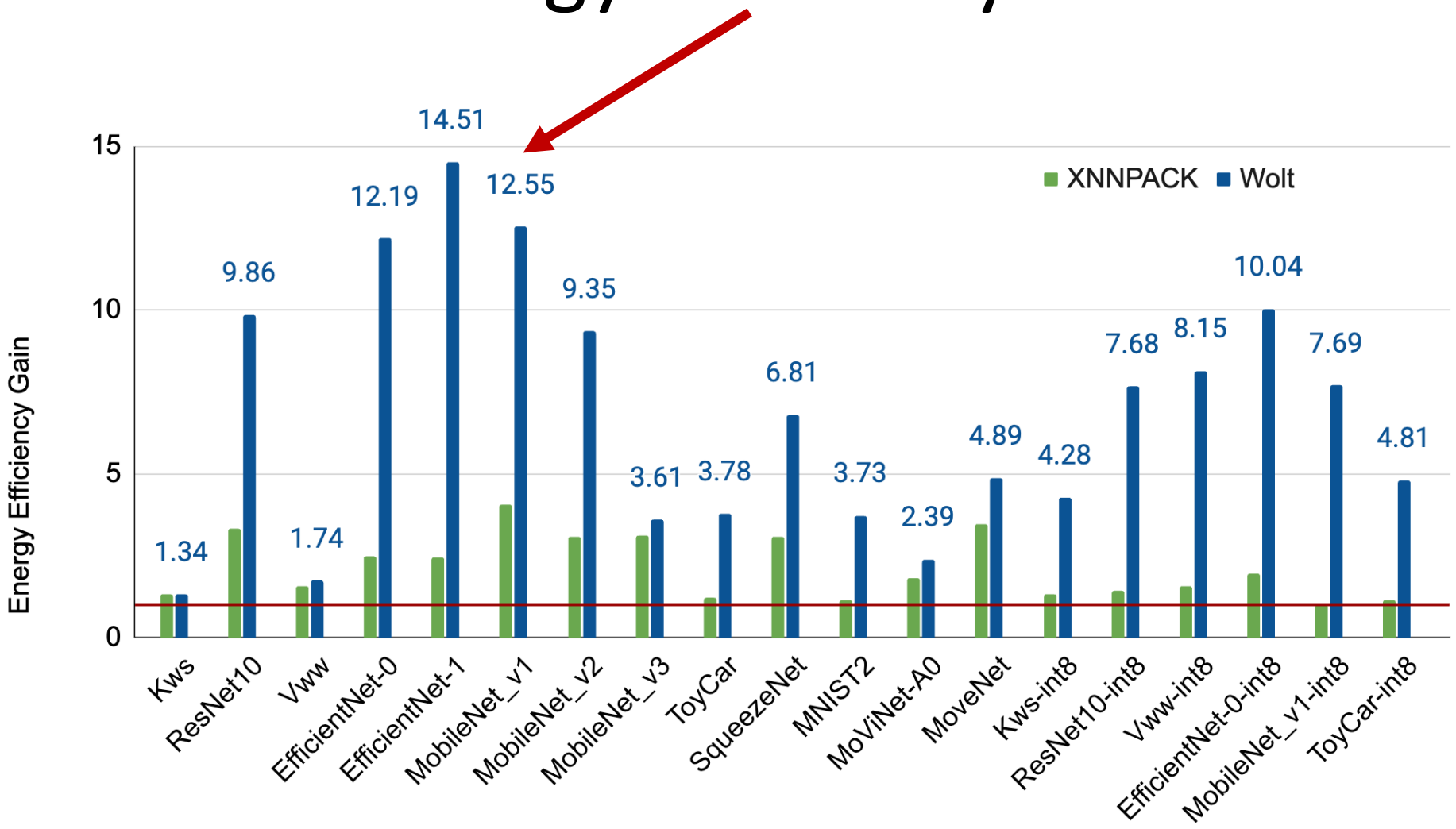

ESP SoC Generator
General SoC configuration:
virtexup
ETH FPnew
No JTAG
Eth (192.168.1.2)
Use SGMII
No SVGA
With synchronizers
Data transfers:
Bigphysical area
Scatter/Gather
Cache Configuration:
Cache En.:
L2 SETS: 512
L2 WAYS: 4
LLC SETS: 1024
LLC WAYS: 16
ACC L2 SETS: 512
ACC L2 WAYS: 4
CPU Architecture:
Core: ariane
NoC configuration
Rows: 2 Cols: 2
Config
Monitor DDR bandwidth
Monitor memory access
Monitor injection rate
Monitor router ports
Monitor accelerator status
Monitor L2 Hit/Miss
Monitor LLC Hit/Miss
Monitor DVFS
Num CPUs: 1
Num memory controllers: 1
Num I/O tiles: 1
Num accelerators: 0
Num CLK regions: 1
Num CLKBUF: 0
VF points: 4
NoC Tile Configuration
(0,0)
mem
(0,1)
cpu
(1,0)
empty
(1,1)
IO
Has L2
Clk Reg:
Has PLL
CLK BUF
Generate SoC config

Evaluation: Performance

- Performance evaluation of Wolt with several different ML models
  - XNNPACK is the state-of-the-art optimized 3rd party software library for NN inference
  - The baseline is the software execution on a single RISC-V core

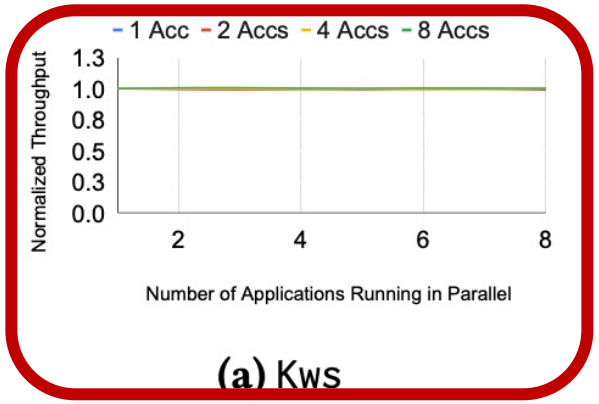


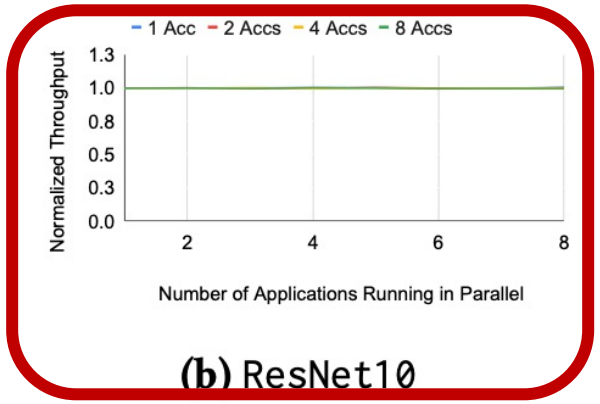
Evaluation: Energy Efficiency Gain

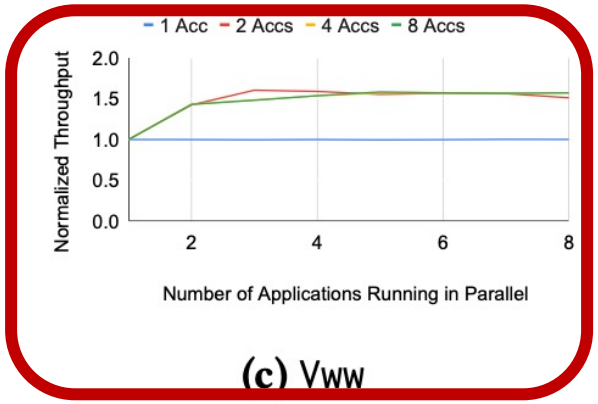


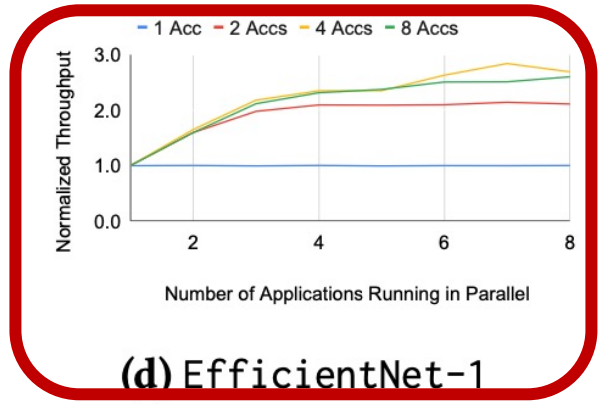
Experimental Evaluation: Wolt Resource Manager

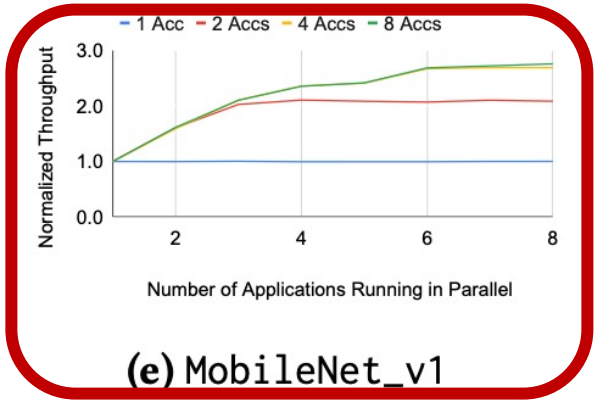
- Evaluation of the resource manager with 8 different ML models

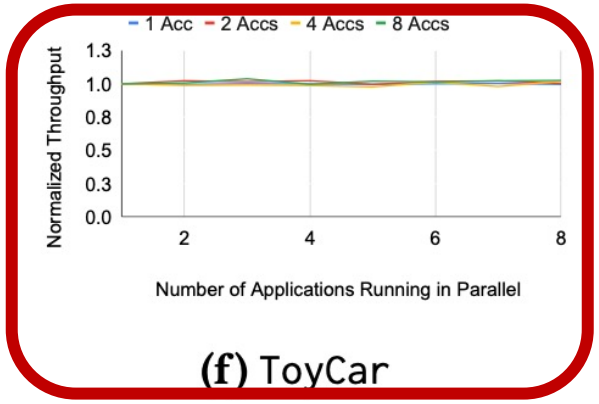


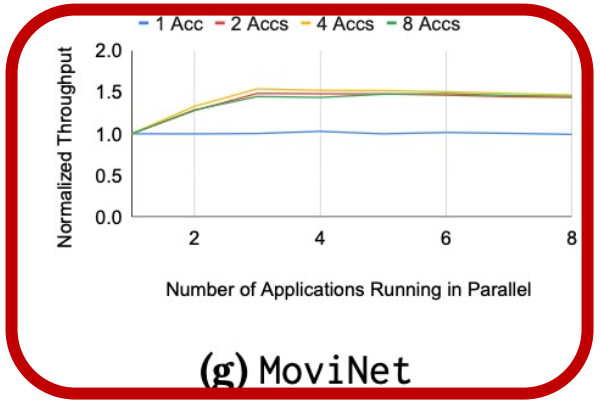


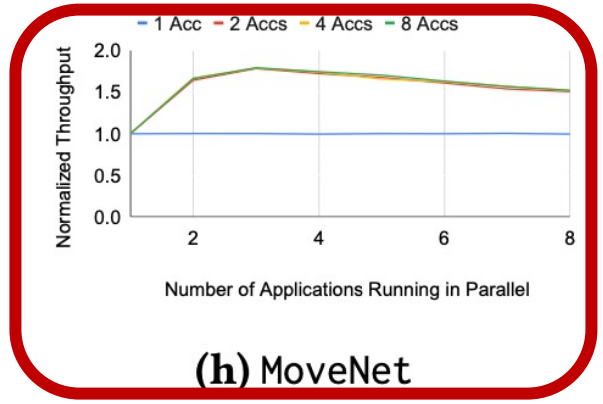












Time duration to run on the CPU > Time duration to run on the accelerators

Applications in Parallel with 2 CPUs

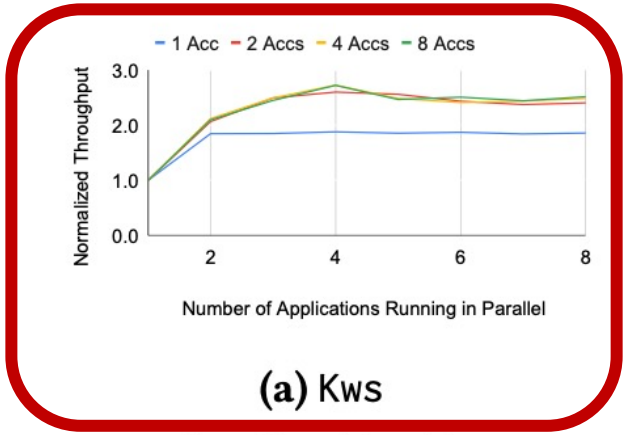
Amdahl's Law:

$$speedup = \frac{S + P}{S + \frac{P}{N}}$$

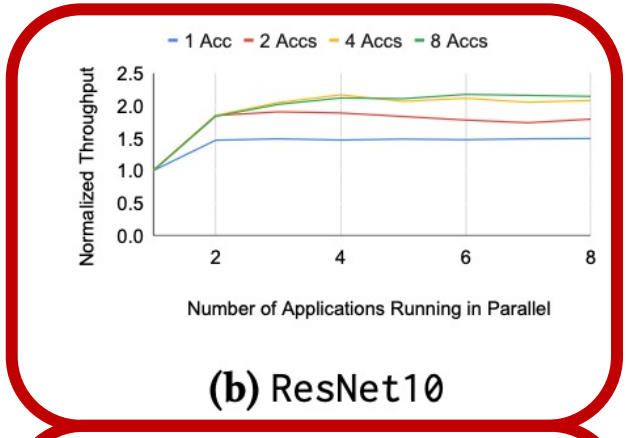
P: fraction can be parallelized
S: fraction that is serial
N: number of CPUs

$$speedup = \frac{S + P}{\frac{S}{M} + \frac{P}{N}}$$

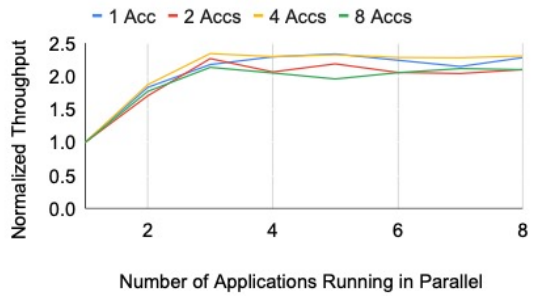
P: time duration on CPU
S: time duration on ACC
N: number of CPUs
M: number of ACCs



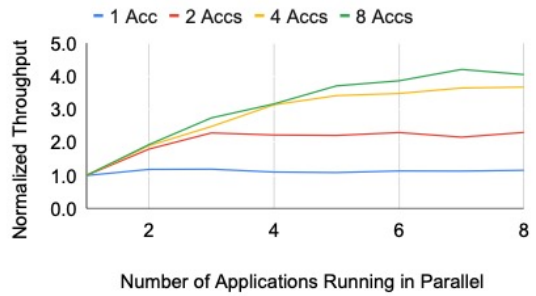
(a) Kws



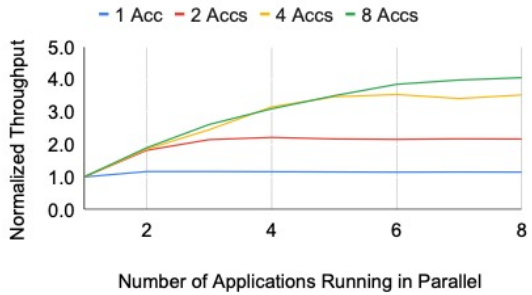
(b) ResNet10



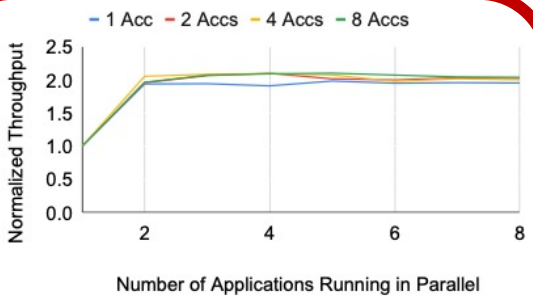
(c) Vww



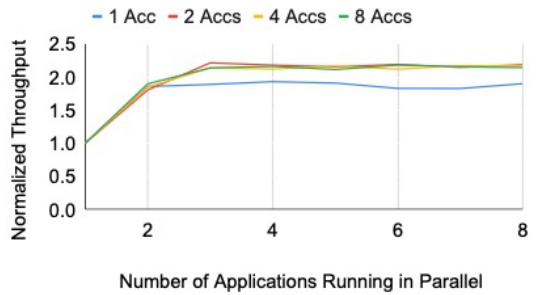
(d) EfficientNet-1



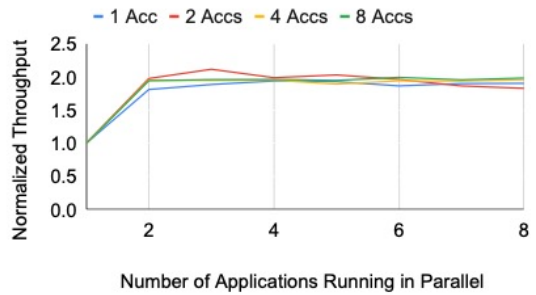
(e) MobileNet_v1



(f) ToyCar



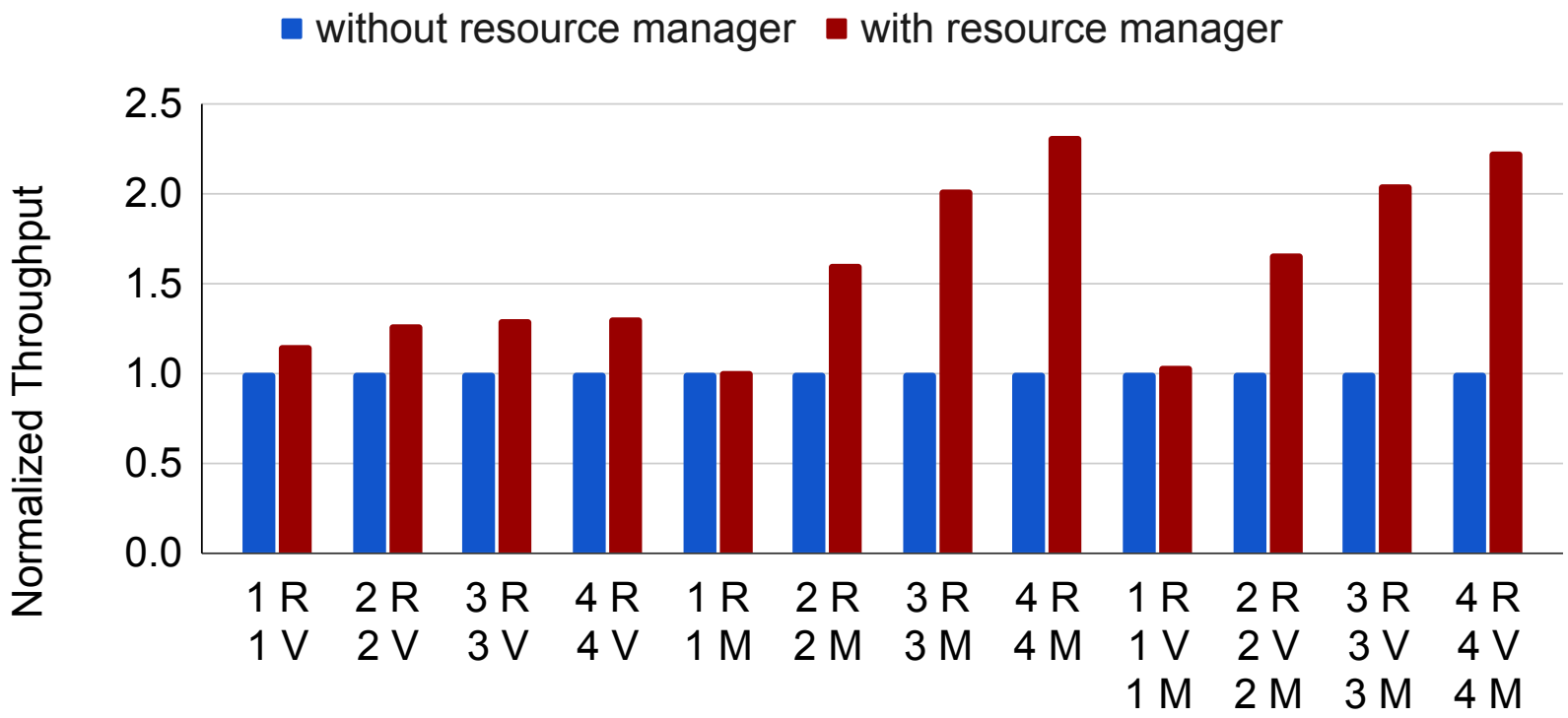
(g) MoviNet



(h) MoveNet

Experimental Evaluation: Multi-Tenant Scenario

- Running a combination of ResNet10 (R), Vww (V), and MobileNet_v1 (M)
- SoC configuration: one CPU core and four Conv2d accelerators

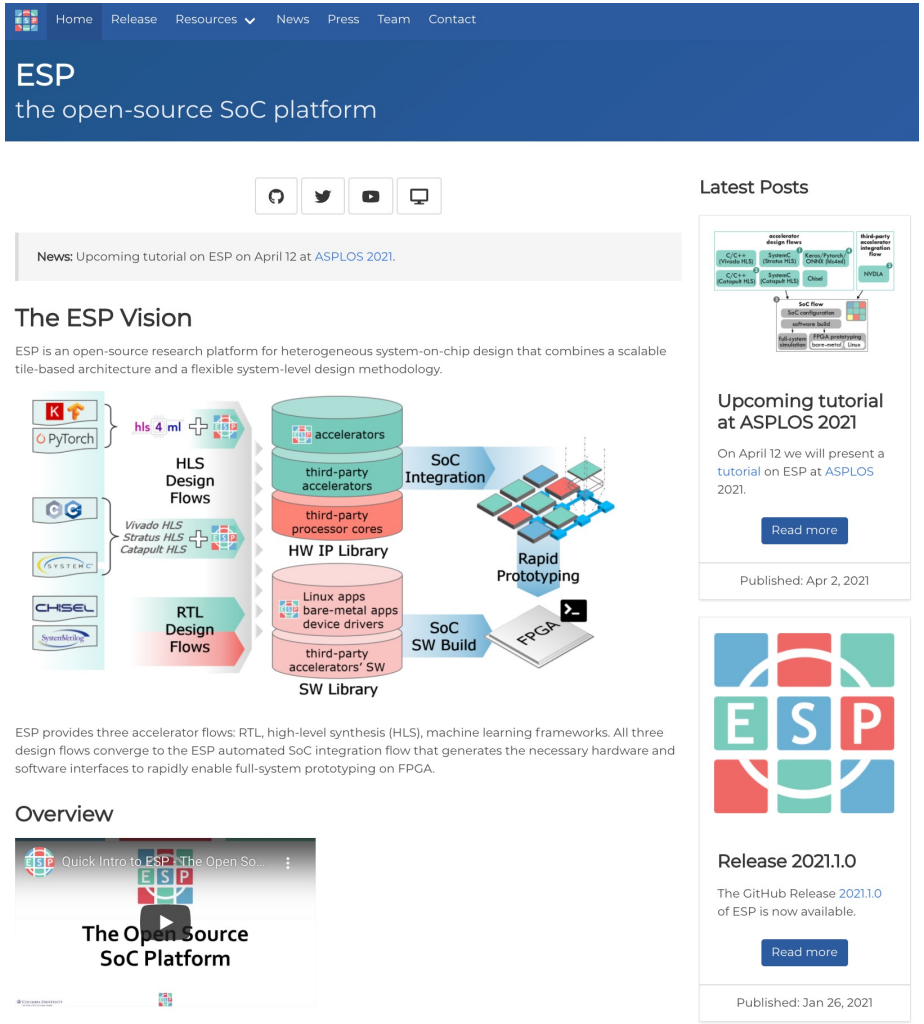


Conclusion and Future Work

- With Wolt, we developed
  - a transparent software layer for TensorFlow Lite ML applications that decouples the development of software and hardware
  - a resource manager for running multi-tenant ML applications that enables the auto-assignment of accelerators and improves the performance for parallel executions

- We plan to
  - extend the Wolt's support to other ML frameworks (e.g., PyTorch and TVM)
  - improve the resource manager by adding more sophisticated scheduling mechanisms and supporting dynamic delegation of a given functionality to different accelerator instances that offer different design trade-offs

ESP : An Open-Source Platform for SoC Design



Tutorials

ISCA 2024 tutorial
The ESP Approach to Agile Chip Design

Latest update: 2024-06-20

- Logistics
- Tutorial Overview
- Preliminary setup
- Program
- Video recording
- Team
  - Speakers

- 9:00 am Jun 30, 2024
- Location: Quebracho A

www.esp.cs.columbia.edu

Thank you from the ESP team!

sld.cs.columbia.edu


ColumbiaSld

esp.cs.columbia.edu


sld-columbia/esp


c/ESP-platform

Transparent Deployment of TFLite Workloads on Lightweight Many-Accelerator Architectures

Kuan-Lin Chiu, Guy Eichler, Chuan-Tung Lin

Giuseppe Di Guglielmo and Luca Carloni