Optimization of AI SoC with Compiler-assisted Virtual Design Platform

Chih-Tsun (CT) Huang[1,2], Juin-Ming Lu[1,2],
Yao-Hua Chen[2], Ming-Chih Tung[2], and Shih-Chieh Chang[1,2]

National Tsing Hua University[1]

國立清華大學

Industrial Technology Research Institute[2]

ITRI

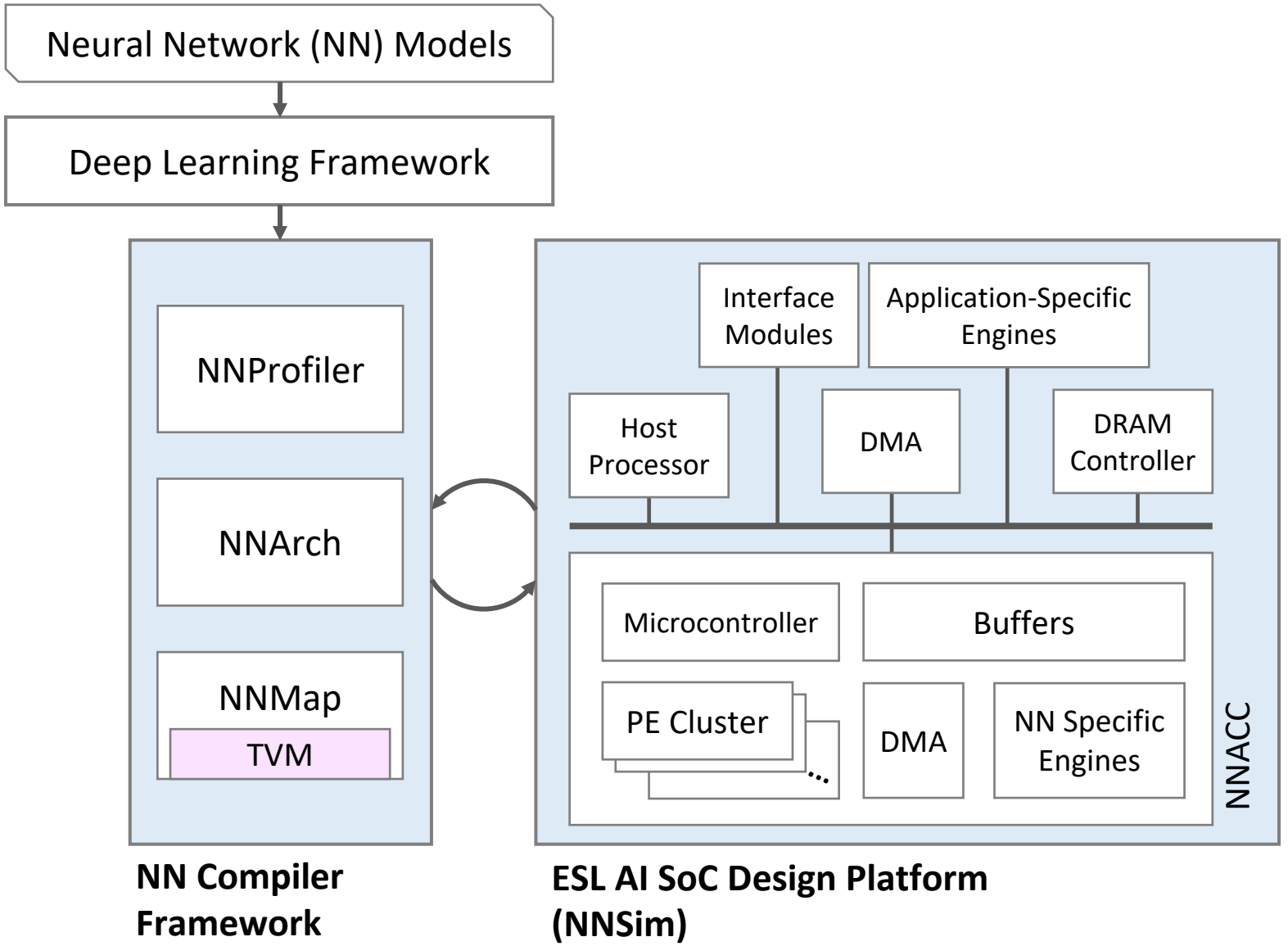
Introduction

- Deep learning keeps evolving dramatically
- Demand for efficient hardware accelerators has become vital
  - Scalable hardware architecture
  - Diversified neural network models
- Lack of software/hardware co-development toolchains
  - Designing efficient AI SoCs (artificial intelligent system-on-chips) is considerably challenging
  - Considering performance, area, energy consumption trade-offs
  - Minimizing gap between algorithms and hardware designs

Compiler-assisted Virtual Design Platform

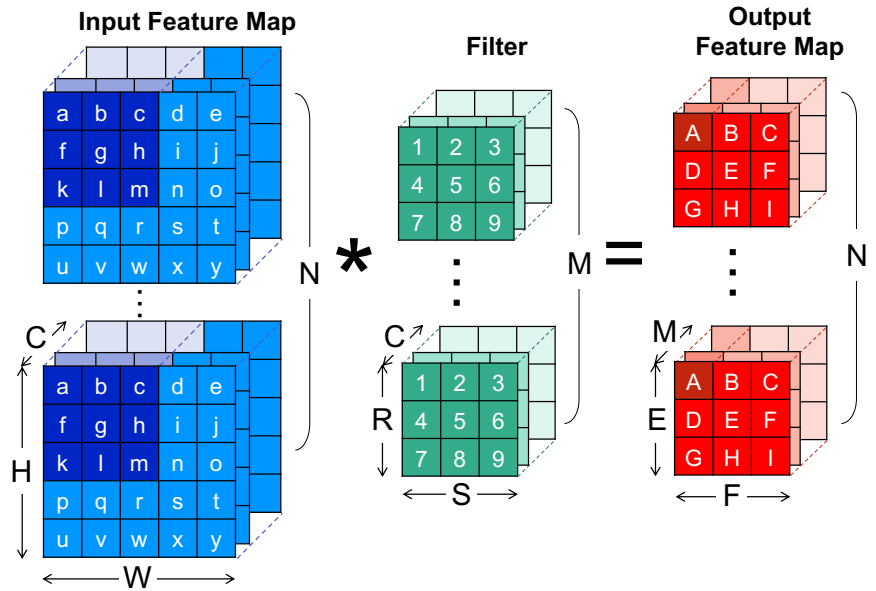
- Virtual Design Platform of AI SoCs
  - Electronic System-Level (ESL)
    - Design and optimize AI SoCs at a high level of abstraction in SystemC (C++)
    - System profiling and performance tuning
  - Enabling rapid design prototyping with high-level synthesis
  - Design exploration and architectural optimization
    - Fast and accurate performance/area/energy estimation
- AI Compiler toolchain assisted
  - Optimized execution of deep learning models on hardware platform
    - Code optimization
    - Parallelization
    - Scheduling
  - SW/HW co-optimization

Our Virtual Design Platform



Convolutional Layer:
Tiled-based Row-Stationary Dataflow

Naïve 7 For-loops Convolution




```
for (n=0; n<N; n++) {
for (m=0; m<M; m++) {
for (f=0; f<F; f++) {
for (e=0; e<E; e++) {
for (r=0; r<R; r++) {
for (s=0; s<S; s++) {
for (c=0; c<C; c++) {
    ofmap += ifmap x filter;
}}}}}}}
```

Tile-based Row-Stationary Dataflow

```
for (n0=0; n0<N; n0+=n) {
for (e0=0; e0<E; e0+=e) {
for (g0=0; g0<G; g0+=g) {
for (m0=0; m0<M; m0+=m) {
for (c0=0; c0<C; c0+=q*r) {
for (m1=0; m1<m; m1+=p*t) {
```

Temporal Execution

```
// ------ Processing Pass ------
// PE Array Level
for (g2=0; g2<g; g2++) {
for (c2=0; c2<r; c2++) {
for (m2=0; m2<t; m2++) {
// PE Set Level
for (r3=0; r3<R; r3++) {
for (e3=0; e3<e; e3++) {
```

Tile-Base Mapping

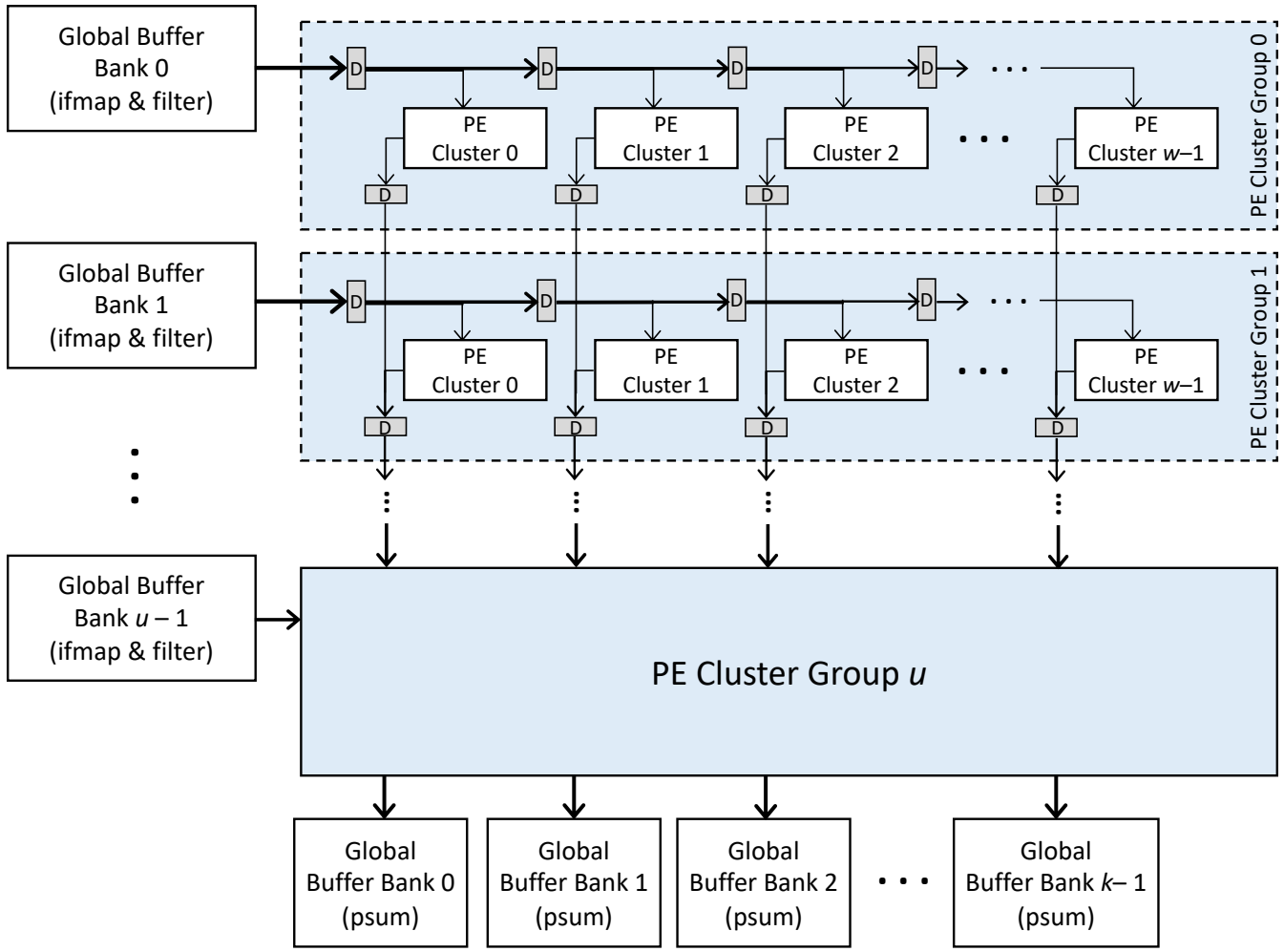
```
// PE Level
for (n4=0; n4<n; n4++) {
for (f4=0; f4<F; f4++) {
for (s4=0; s4<S; s4++) {
for (c4=0; c4<q; c4++) {
for (m4=0; m4<p; m4++) {
    ofmap += ifmap x filter;
}}}}}}}}}}}}}}}}
```

RS 1D Convolution in Each PE

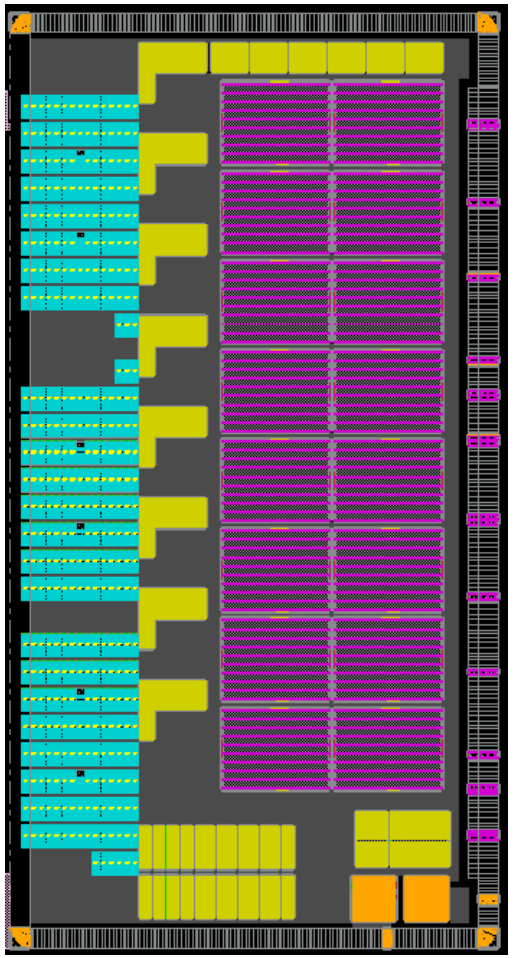
Spatial and Temporal Execution

[VLSI-DAT'23]

NNACC: Tiled-based DCNN Accelerator Architecture

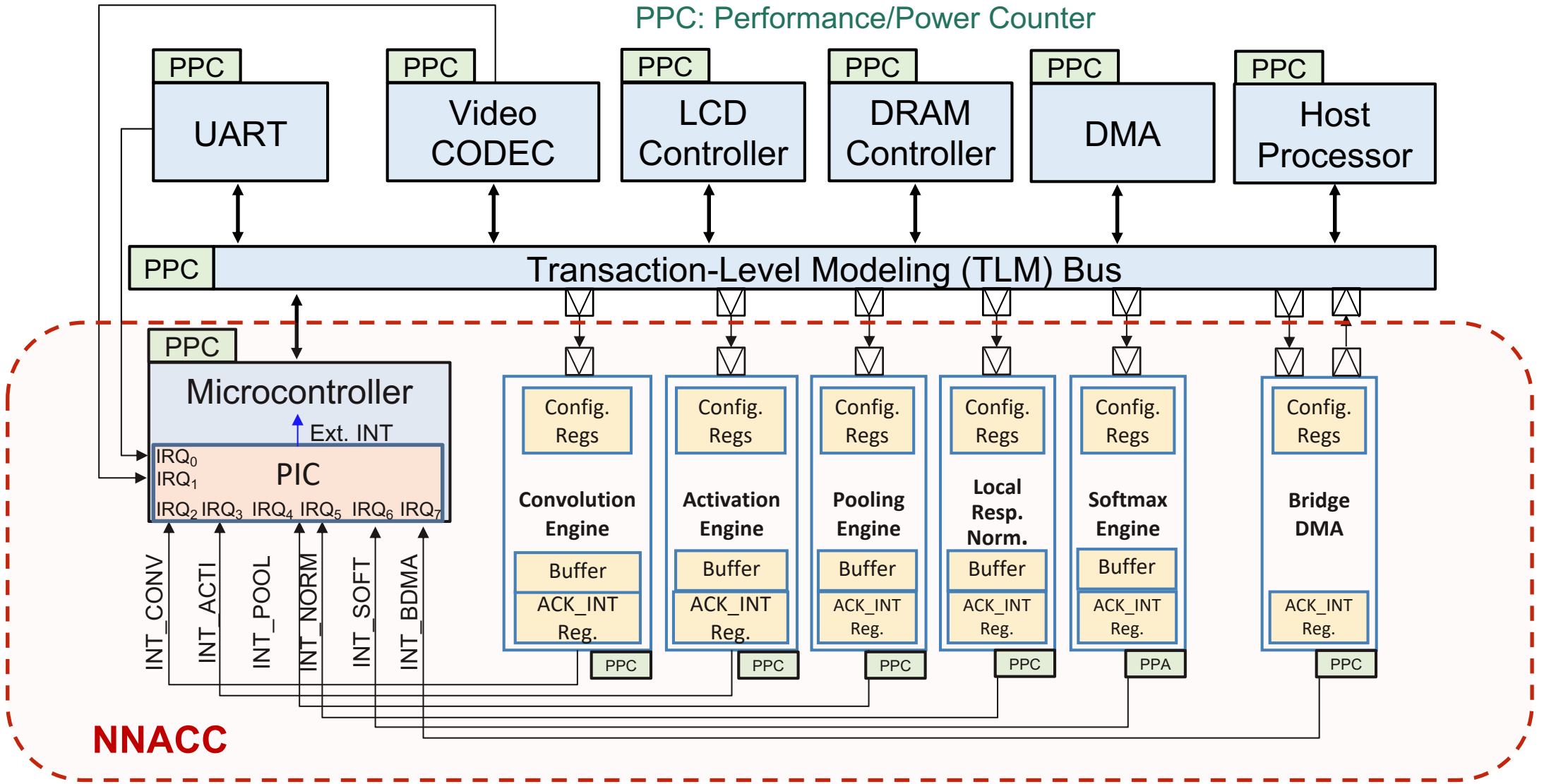


NNACC Test Chip

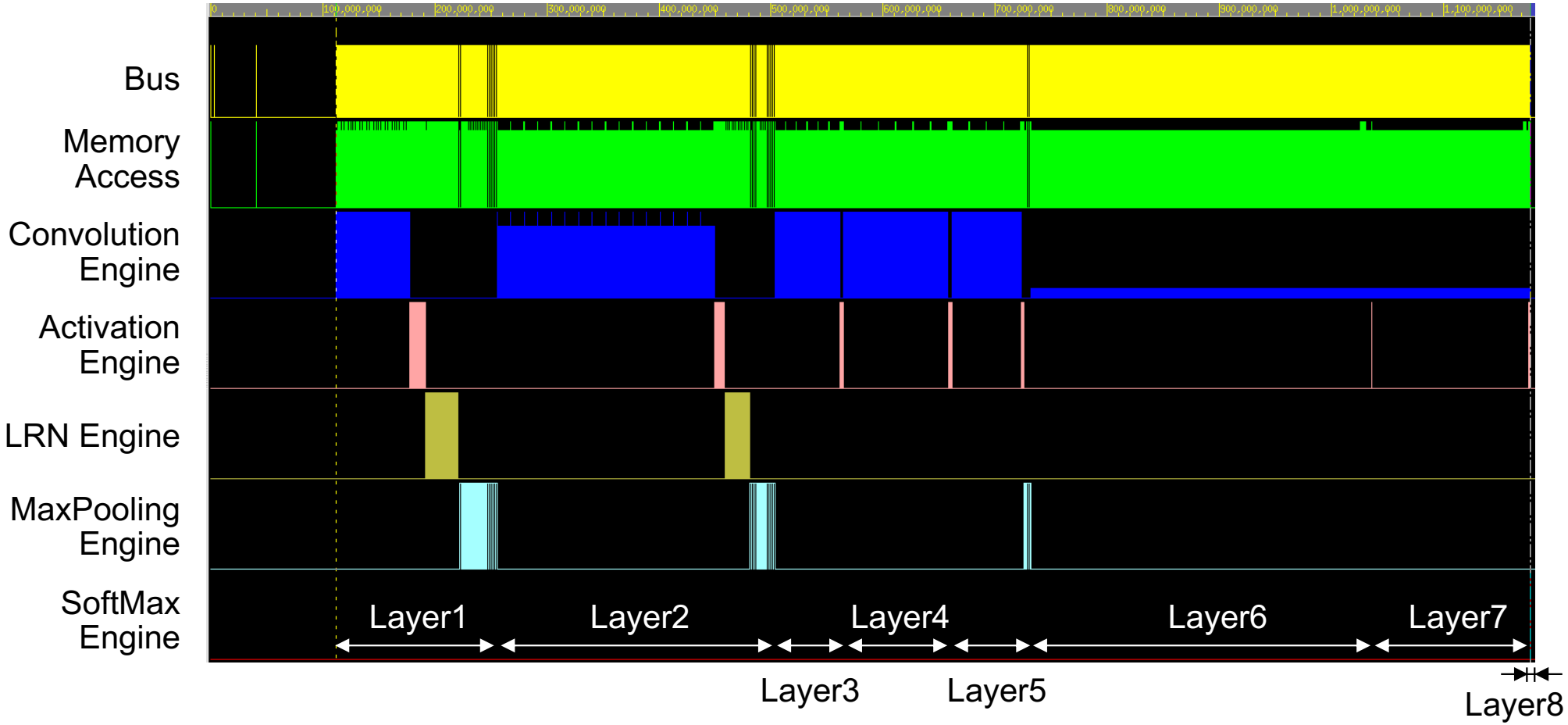


- TSMC 28nm
- RISC-V controller
- 1008 PEs
- ~500KB SRAM
- Integrated with high-bandwidth DRAM
- @1GHz
- 2TOPS for DCNN acceleration

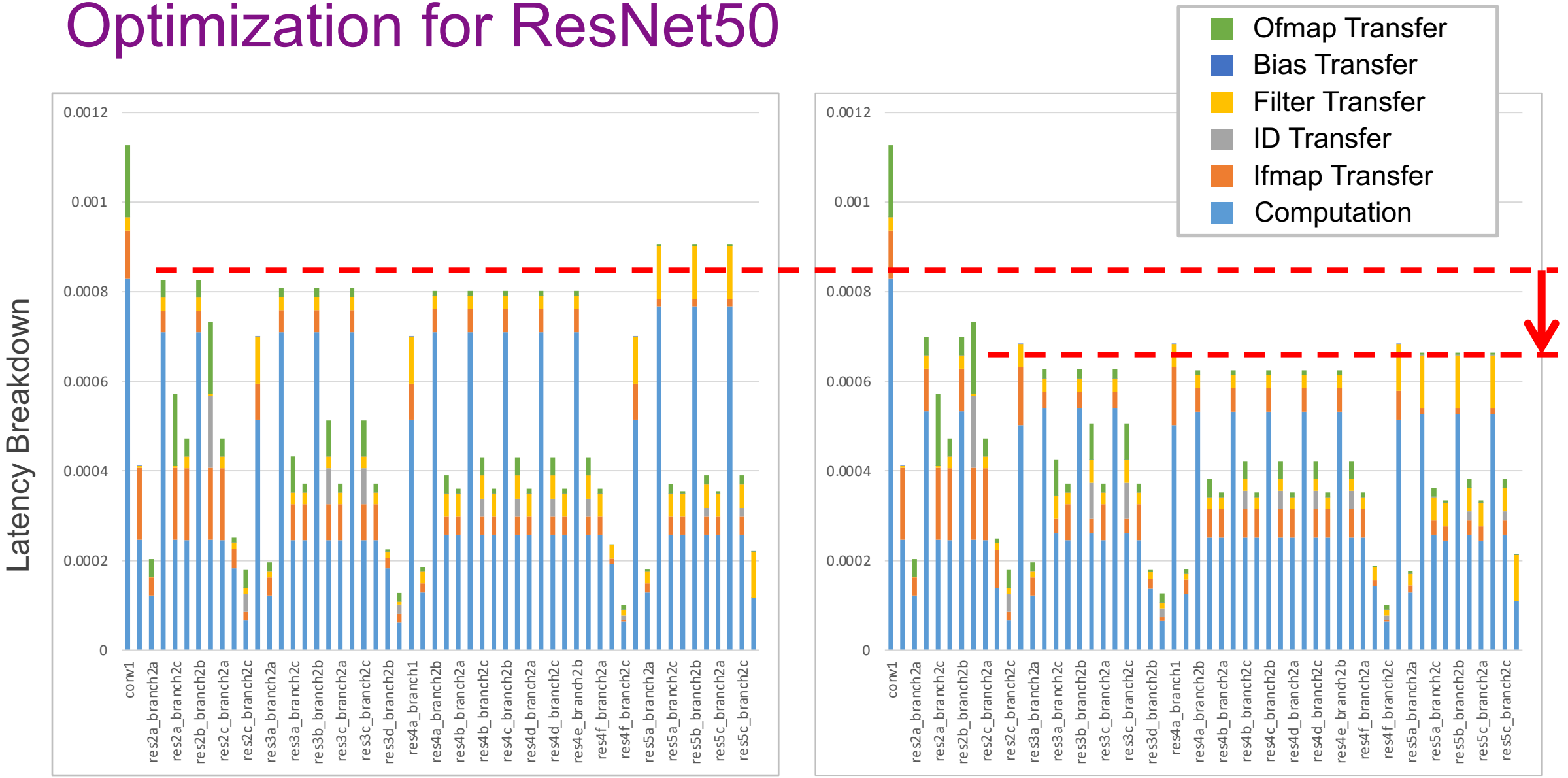
ESL Platform of Deep Inference Processor (DIP)



Example Timing Analysis of AlexNet

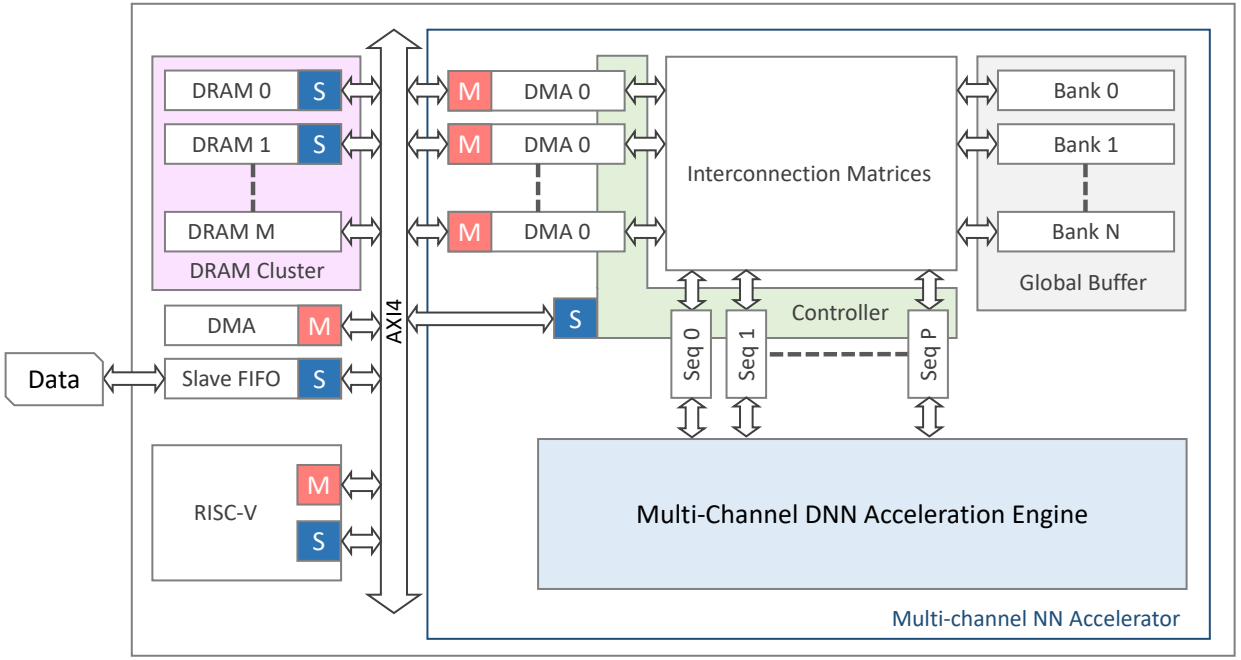


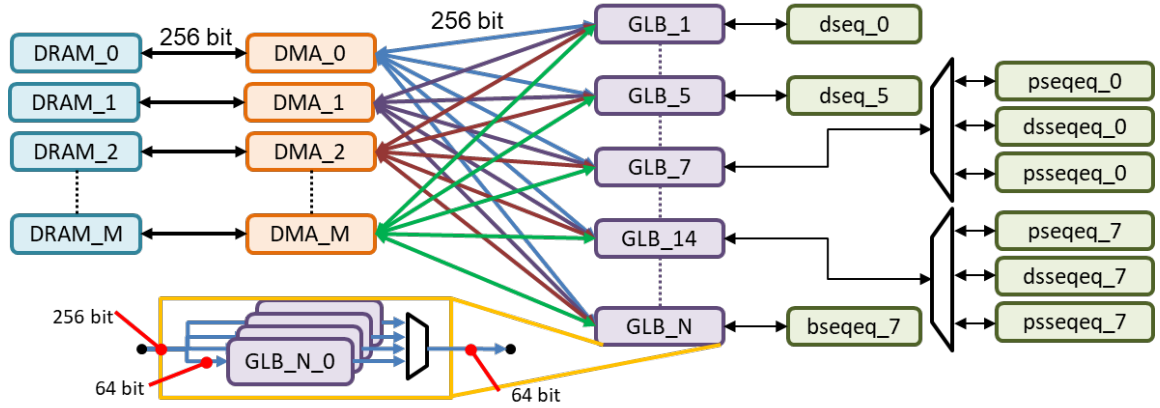
Performance Profiling and Optimization for ResNet50



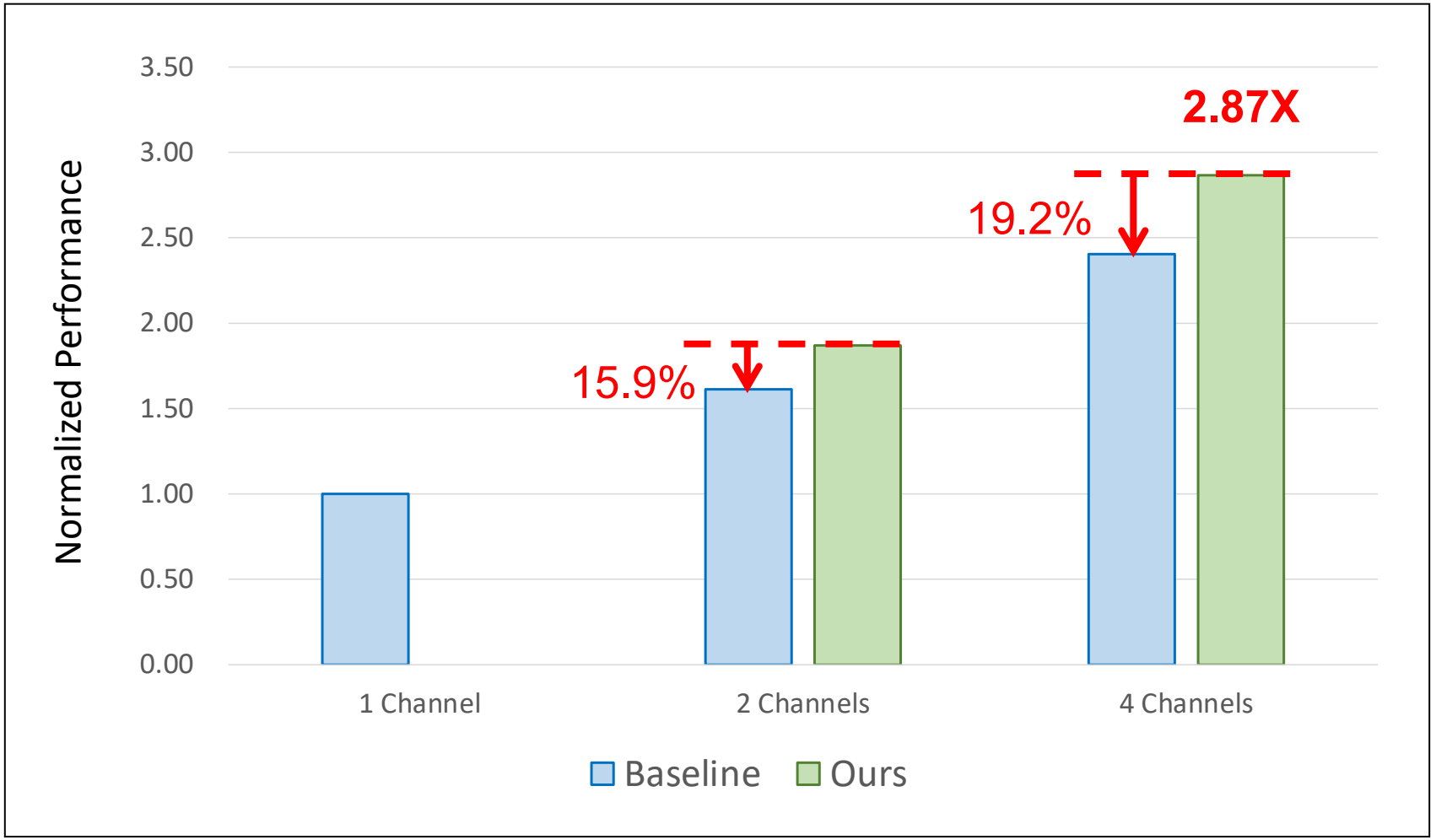
Multi-channel DMA Architecture

- Data movement is essential in DNNs
- Reordering both data and control commands to balance memory transfer

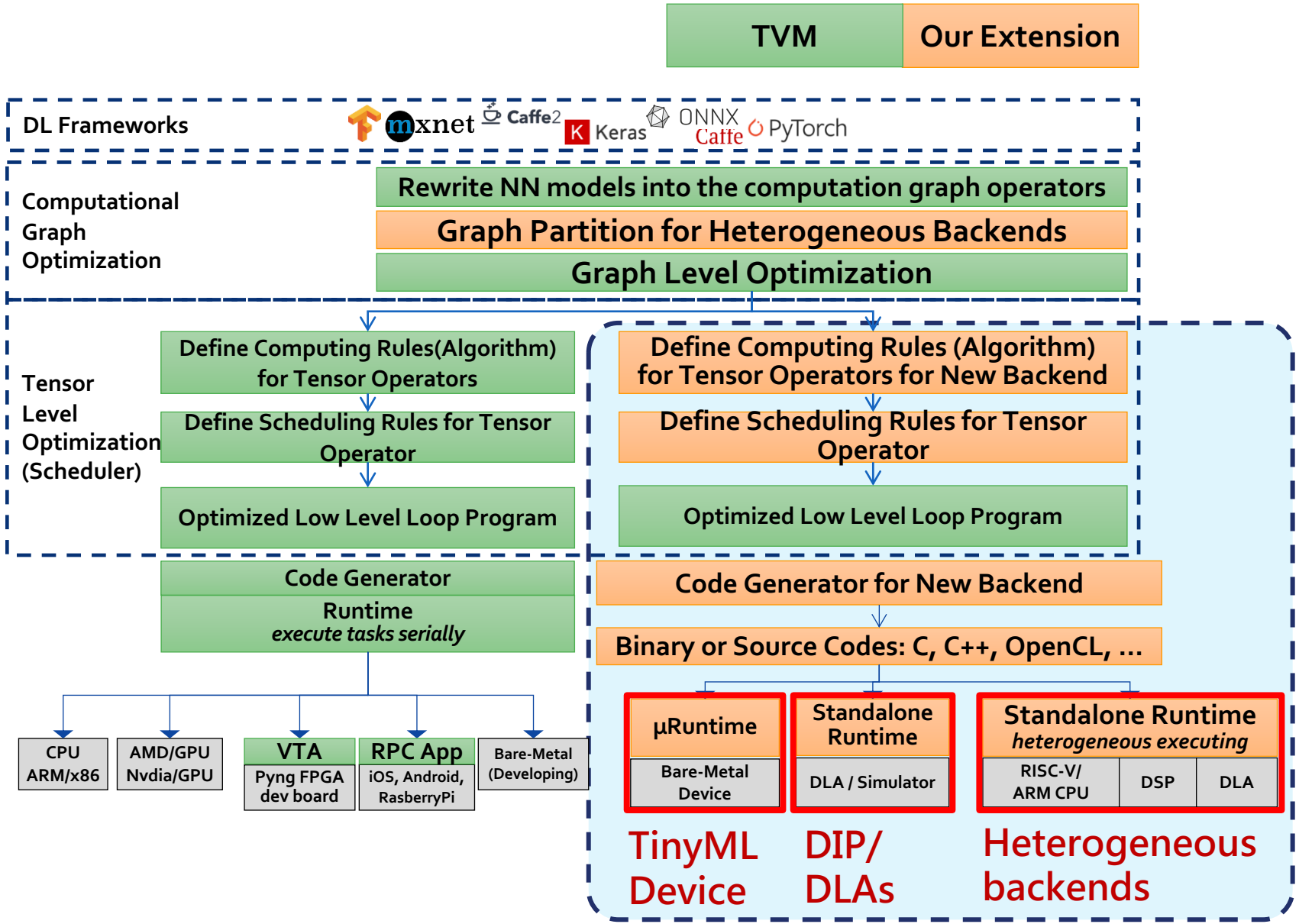




Comparison on Channel Utilization

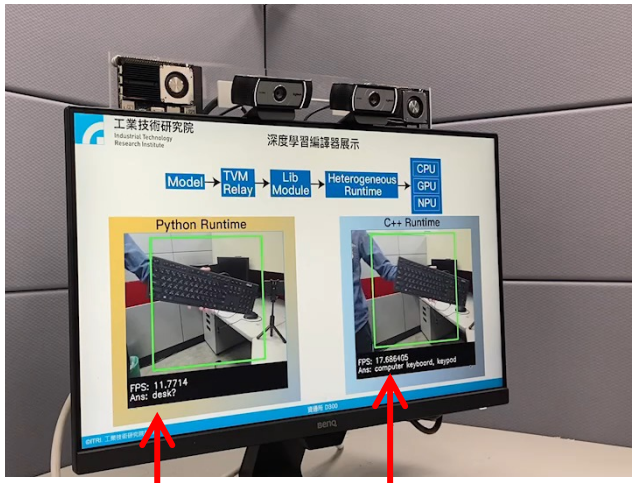


NNMAP: AI Compiler for Heterogeneous Computing



(Khadas)
VIM3P EVB

Heterogeneous Execution on Evaluation Board with CPU+GPU+NPU

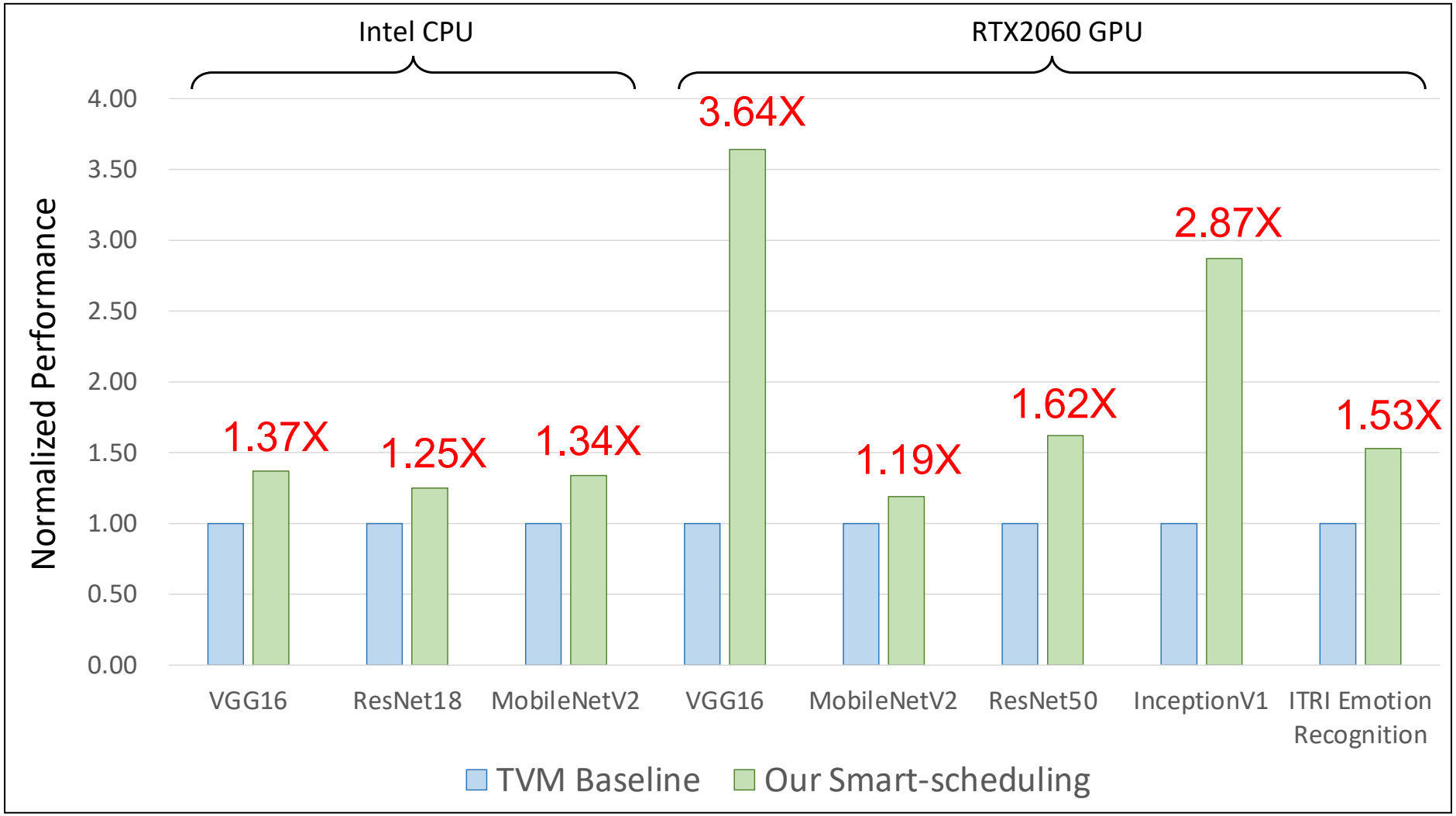


Python Runtime

Our C++ Runtime

Smart-scheduling

- Hardware-aware template auto-gen for scheduling search
- Performance improvement: 1.85X on average



Conclusion

- Compiler-assisted Virtual Design Platform for AI SoCs
  - ESL platform
    - Rapid functional verification
    - Hardware design exploration for performance, power, area
  - AI Compiler
    - Heterogeneous computing for diversified DNNs
    - Smart scheduling