UF Herbert Wertheim College of Engineering
UNIVERSITY of FLORIDA

Coarse-Grained Floorplanning for streaming CNN applications on Multi-Die FPGAs

Danielle Tchuinkou Kwadjo

ECE Department, University of Florida

Outline

1

INTRODUCTION AND MOTIVATION

- Introduction
- DL Evolution
- Challenges
- FINN

2

Proposed framework

- Rale Balancing
- Multi-level Graph Partitioning
- Coarse-grained Floor planning

3

Results

- Granularity Exploration
- Resource Utilization
- Productivity

4

Conclusion

- DNN workloads have become significant.
- As the complexity of machine learning algorithms increases, more data becomes available.

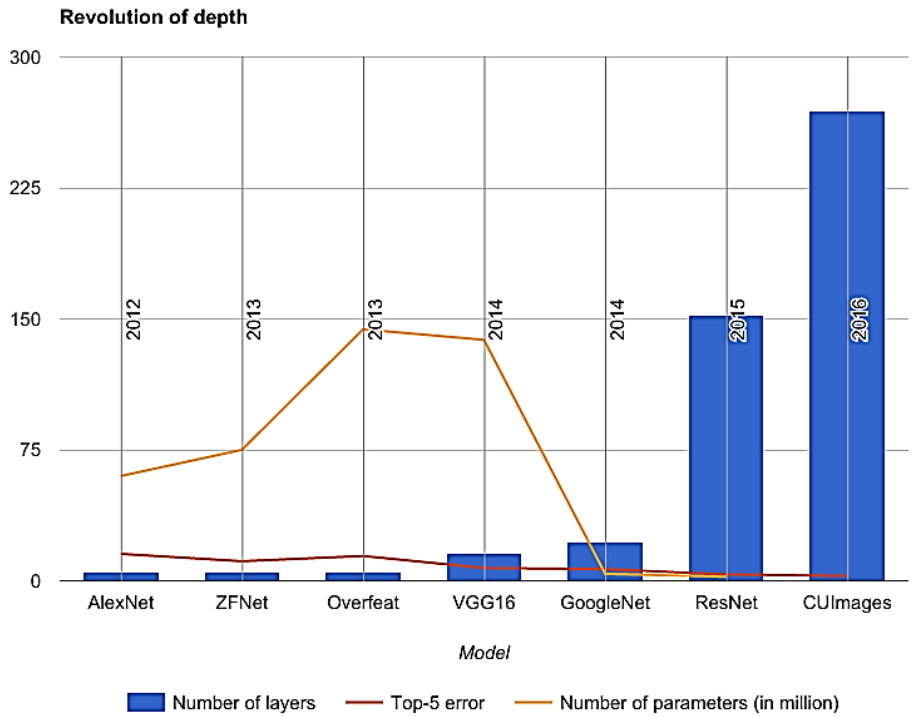


Figure 1: Comparison of parameters size of various compact CNN models

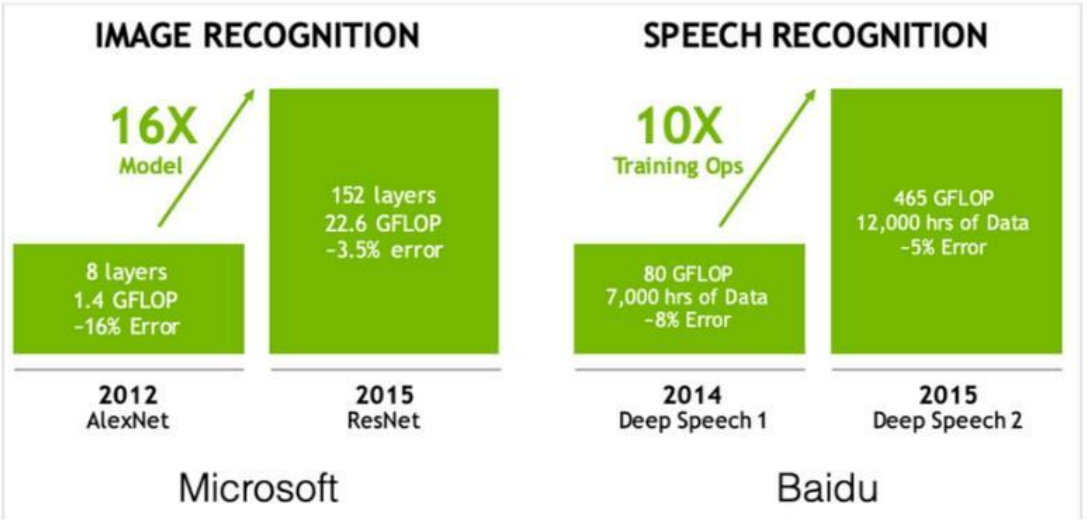


Figure 2: Evolution of depth, error-rate, and number of parameters over the years

FPGA Overview and Benefits

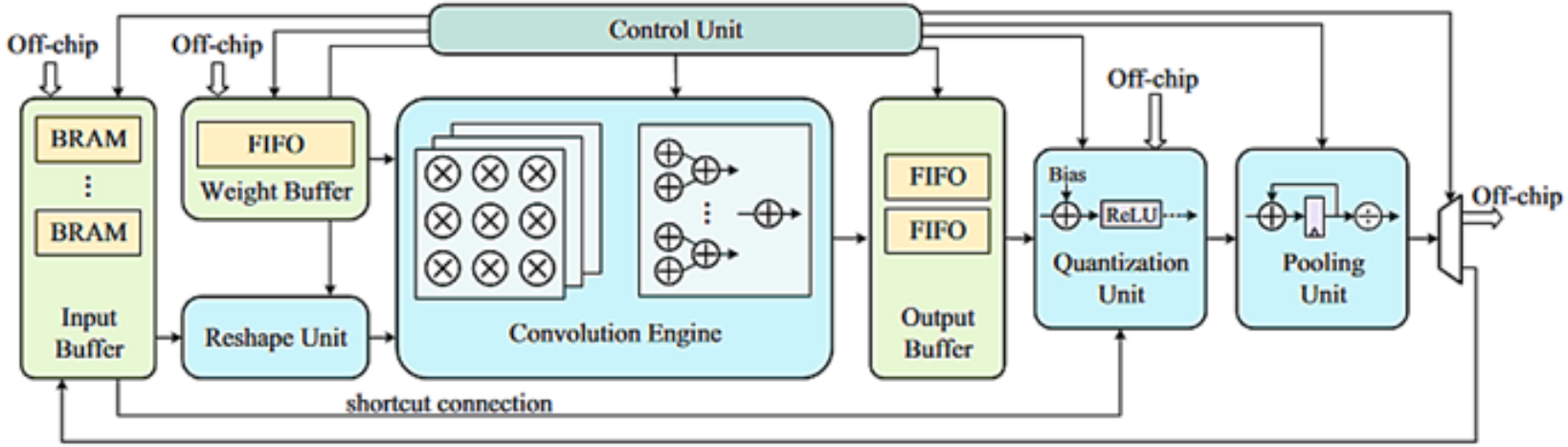
- Both GPU and FPGA are growing fast in artificial intelligence acceleration area.
- GPU is now dominating the market as it has less engineering.
  - Perform poorly when it comes to inference as requests often arrive one at a time.

- However, compared to GPU, FPGA has several outstanding features:
  - **Flexibility:** FPGA allows engineers to reconfigure underlying hardware architecture.
  - **Efficient** with lower precision deep learning algorithms, such as binary neural network [1] and ternary neural network[2].
  - **Low latency:** FPGAs are capable of data parallelism, but also pipeline parallelism
  - **High Power Efficiency**:
    - Xilinx Virtex Ultrascale+, FPGA board has general purpose compute efficiency of 277 GOP/s/W,
    - NVidia Tesla P4, GPU produced by Nvidia, the efficiency is of 208 GOP/s/W

Figure 4: Xilinx Alveo U50

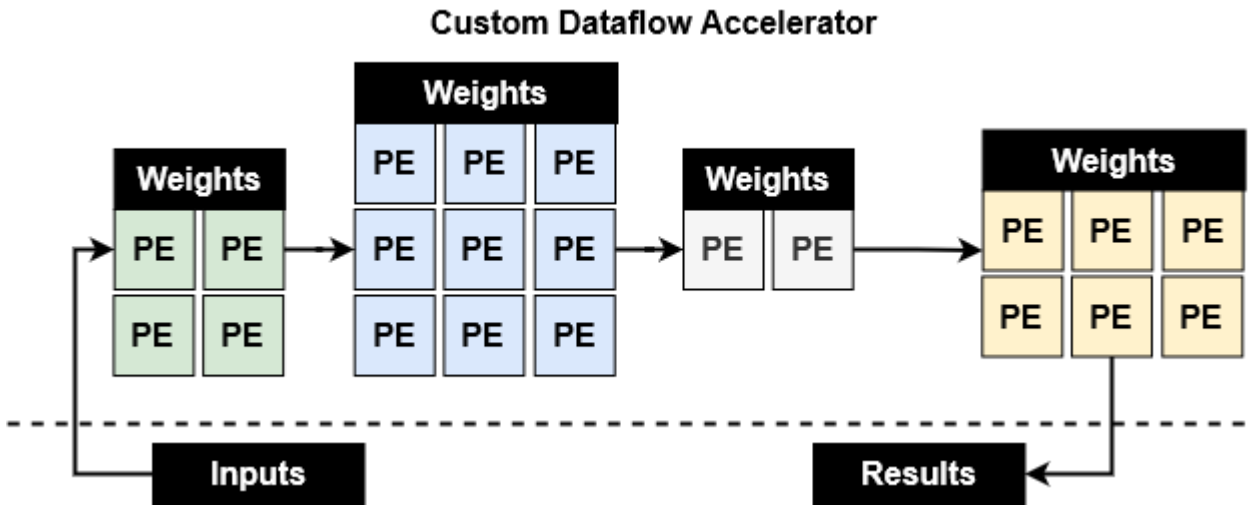
CNN Accelerators

- Most FPGA inference accelerators are based on overlay architectures [1], [2], i.e., GPP MM circuits onto which compute is scheduled to execute the CNN layers in sequence.
  - This approach is flexible, as it enables potentially any CNN topology to be executed by a single accelerator,
  - However, it is not efficient →frequent transfers of weights and activations from/to the memory



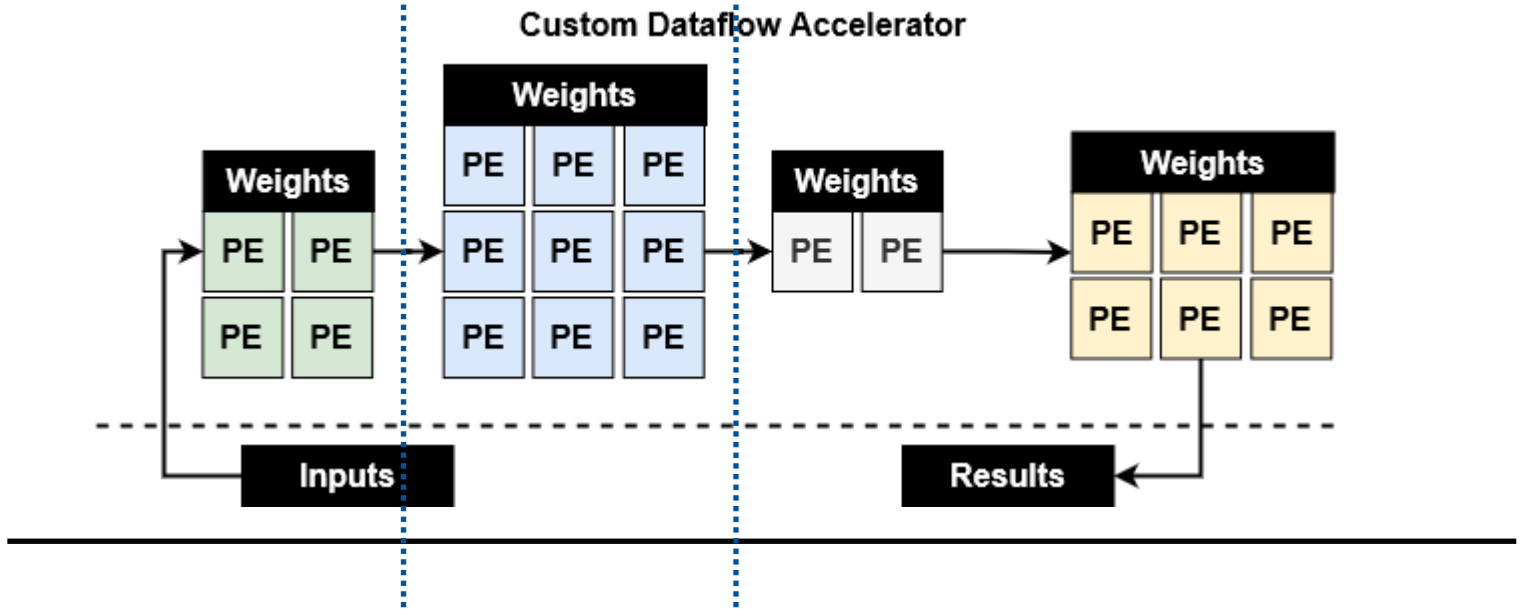
CNN FPGA Accelerators

- An alternative FPGA accelerator architecture is streaming accelerator
  - CNN inference has achieved the lowest latency, highest throughput, and lowest power dissipation.
  - It is limited by the resources on the FPGA.
  - Better fit for cloud environment



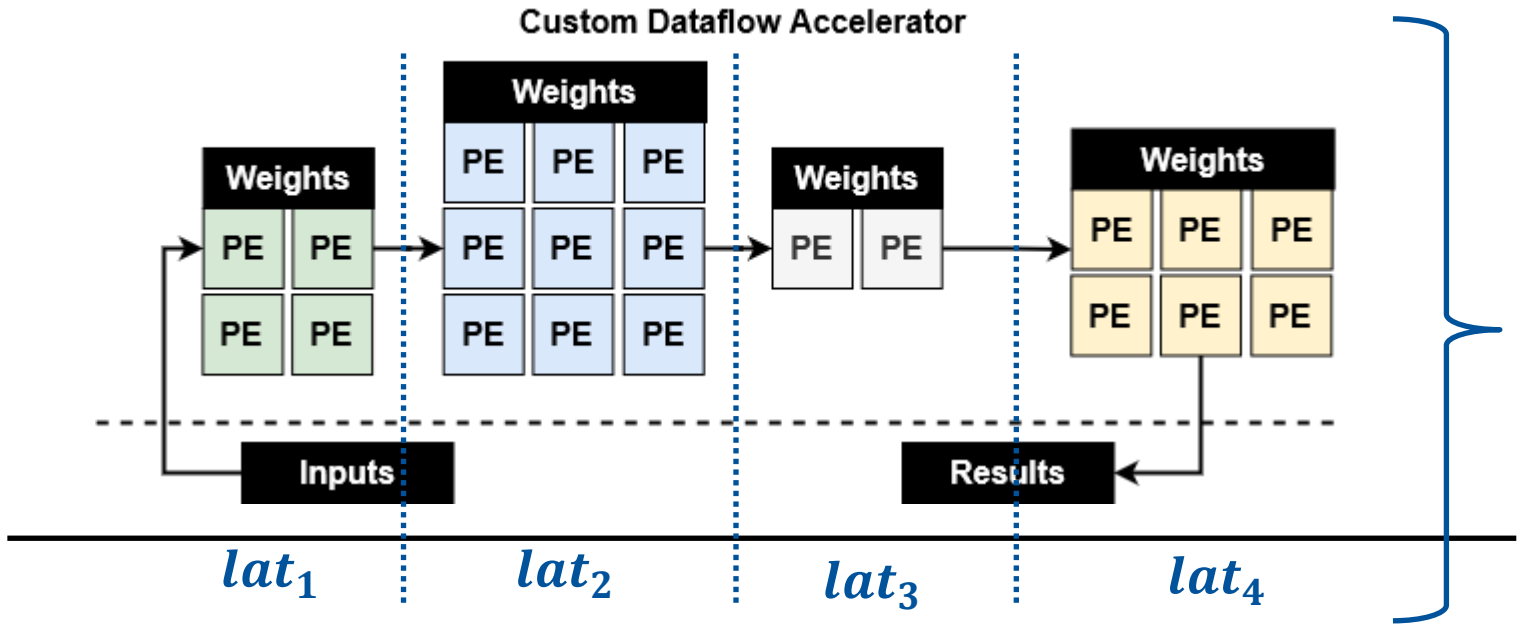
CNN Accelerators Challenges

- Throughput: $T = \frac{\#batch}{\max\{lat_1, lat_2, \ldots, lat_n\}}$ $\Rightarrow$ $lat_1 \backsimeq lat_2 \backsimeq, \ldots, lat_n$



CNN Accelerators Challenges

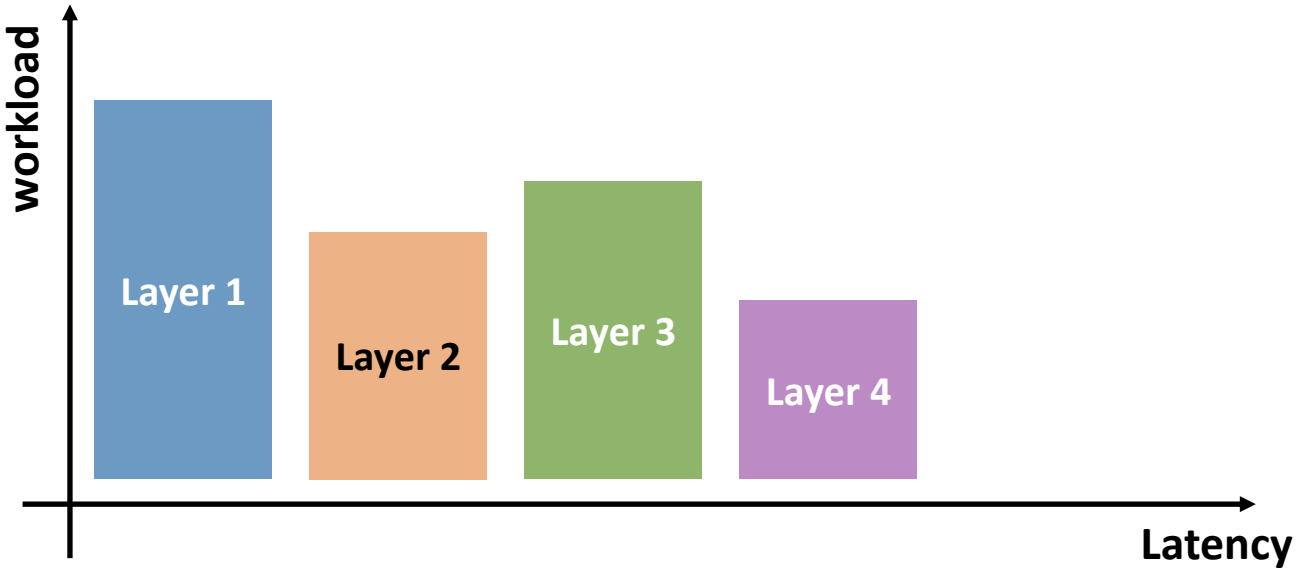
- Throughput: $T = \frac{\#batch}{\max\{lat_1, lat_2, \ldots, lat_n\}}$ → $lat_1 \backsimeq lat_2 \backsimeq, \ldots, lat_n$



- Workload:
  - #PEs?
  - Data processed in parallel

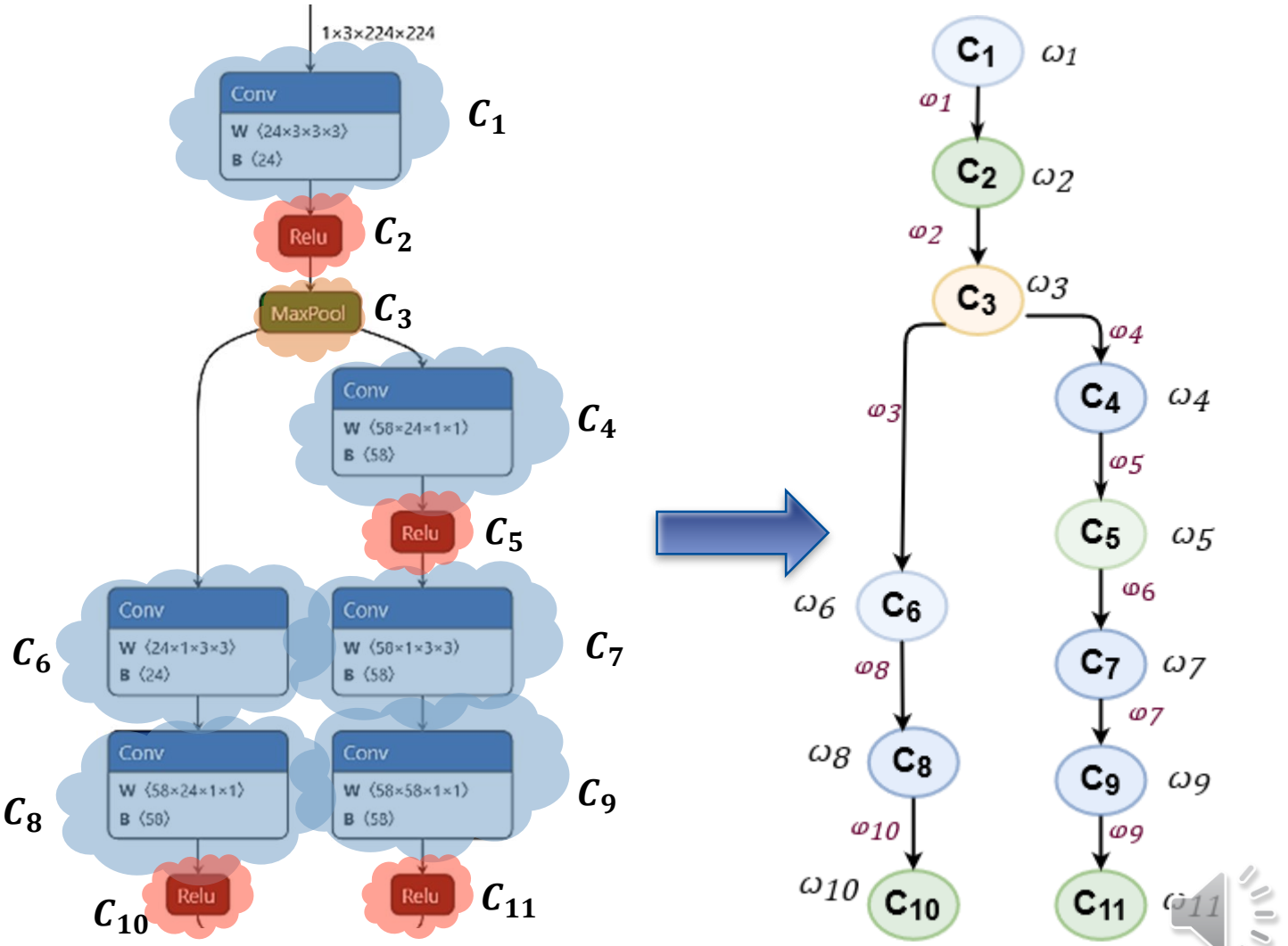
Proposed Framework

1) **Rate Balancing**: we propose a MIP model to find the optimal parameters for a CNN accelerator.



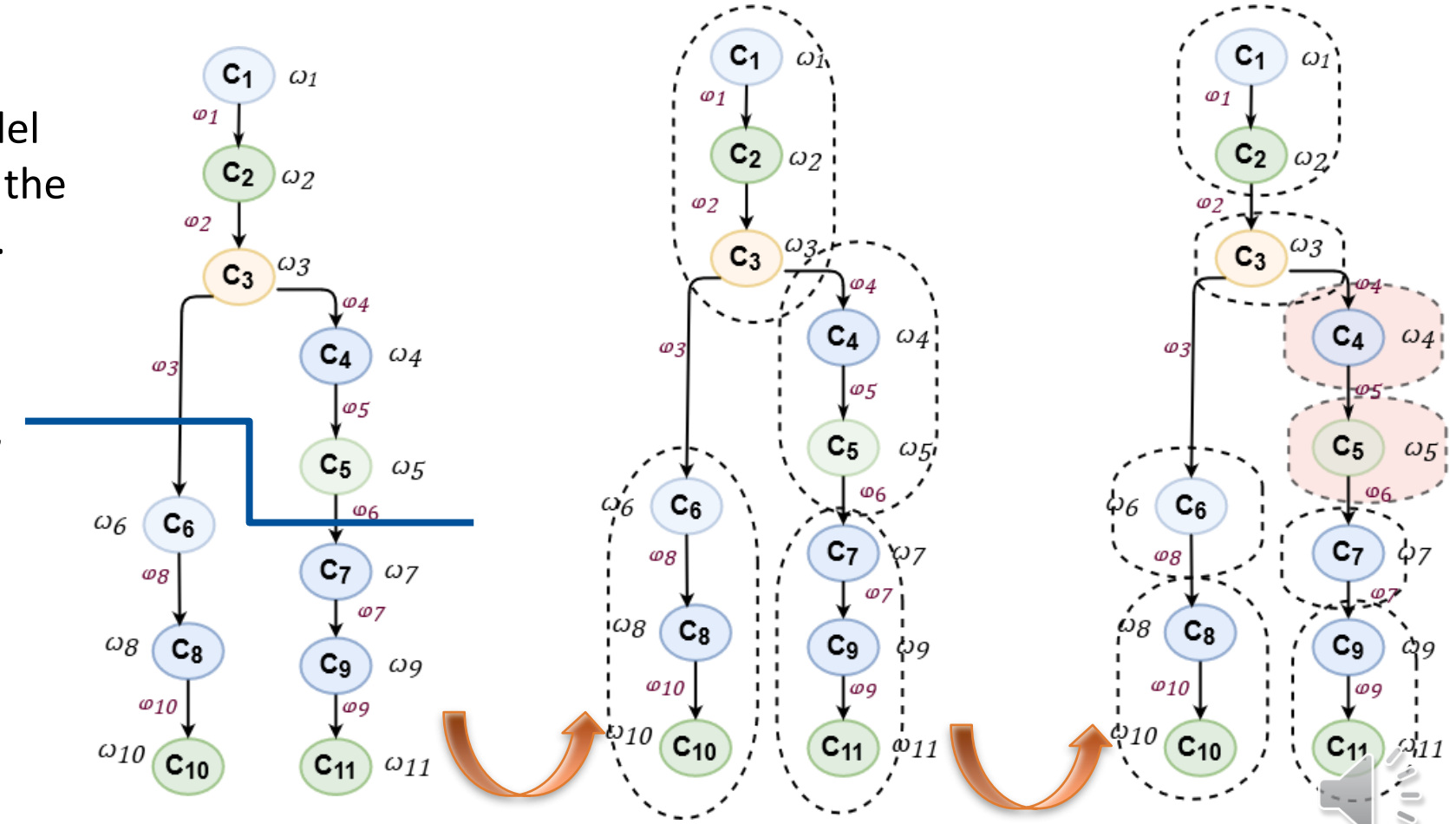
Proposed Framework

1) **Rate Balancing**: we propose a model to find the optimal parameters for the configuration of a CNN accelerator
2) **Computational Graph** $G = (V, E, \omega, \varphi)$
   - set of Vertices V, vertices weights ω,
   - edge set E, edges weight ϕ.



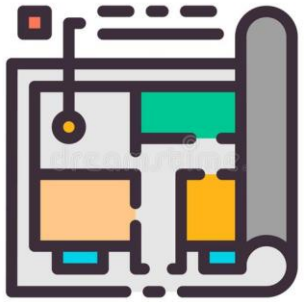
Proposed Framework

1) **Rate Balancing**: we propose a model to find the optimal parameters for the configuration of a CNN accelerator.
2) **Computational Graph** $G = (V, E, \omega, \varphi)$
   - set of Vertices V, vertices weights ω,
   - edge set E, edges weight ϕ.
3) **Multi-level Graph Partitioning**



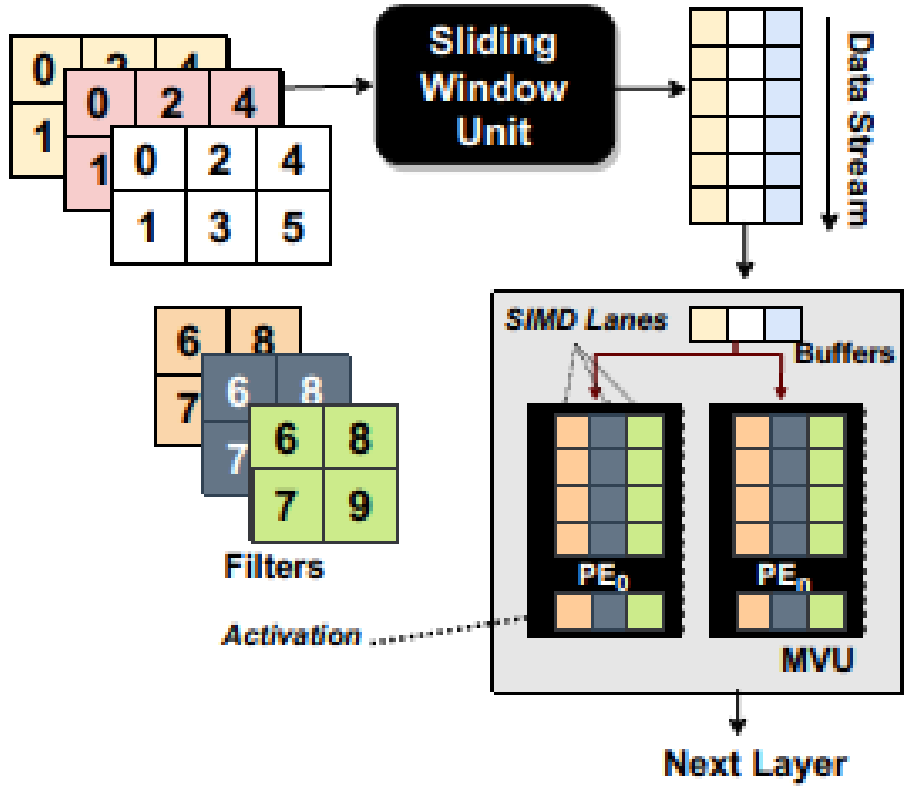
Proposed Framework

1) **Rate Balancing**: we propose a model to find the optimal parameters for the configuration of a CNN accelerator.
2) **Computational Graph** $G = (V, E, \omega, \varphi)$
   - set of Vertices V, vertices weights ω,
   - edge set E, edges weight ϕ.
3) **Multi-level Graph Partitioning**
4) **Coarse-grained Floor planning**



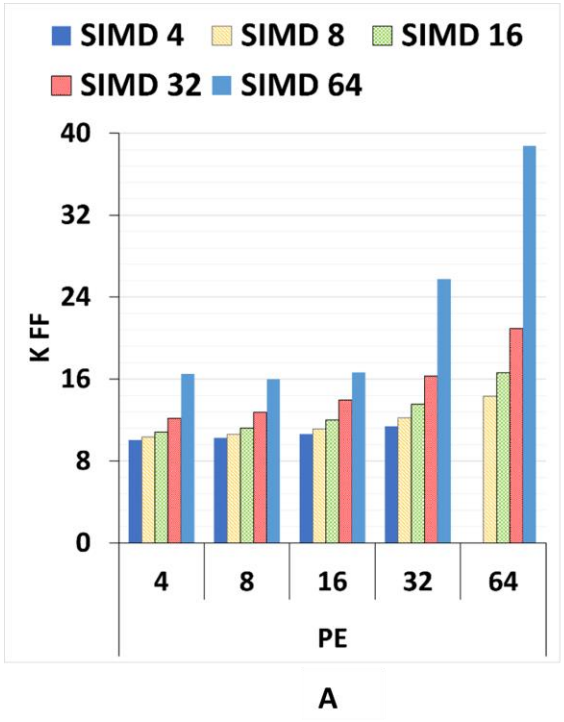
FINN

1) FINN enables the design of heterogeneous custom streaming architecture for a given topology
2) Separate compute engines are dedicated to each layer, communicating via on-chip data streams.
3) It has two main units:
   1) The **Sliding Window Unit** (SWU): Supplies the MVU with the image by applying interleaving and implementing the im2col algorithm.
   2) The **matrix-vector unit** (MVU): input and output buffers and an array of PEs., each with several SIMD lanes.

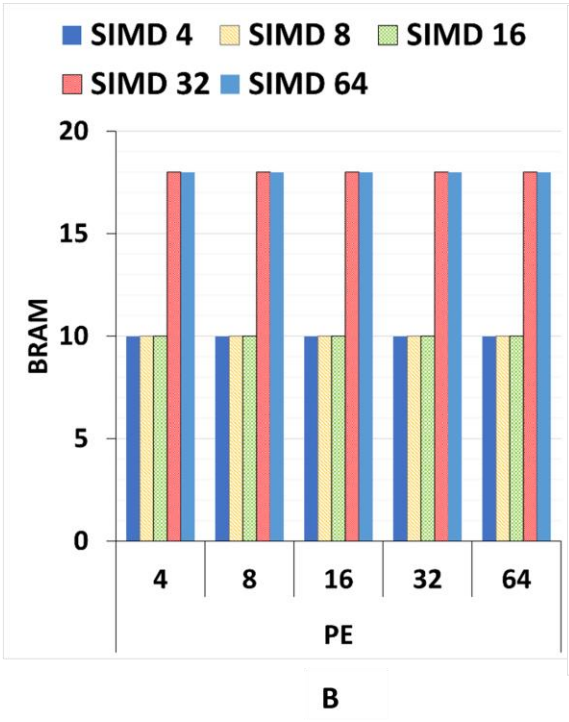


FINN

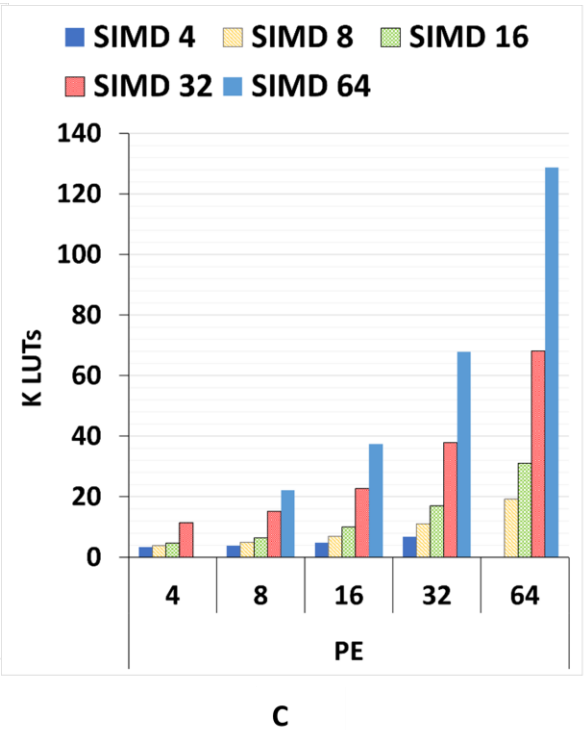
- Convolution Kernel: 3x3, IFM: 56 x56, C=64, OFM 56 x56



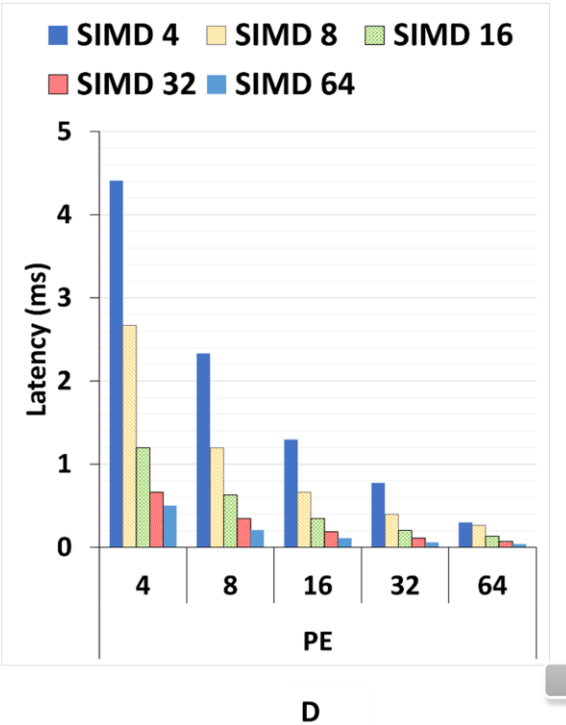
A



B



C



D

Details of the proposed framework

1) Inference model with N vertices and a platform with M SLRs,
   - For a layer i → Maximize (S_i, P_i) such that

$$\begin{cases} Latency_i \simeq (1+\epsilon) \times Latency_{i+1} \quad \forall i = 1, ..., N \\ with\ Latency_i = F_i^n \times F_i^s \\ with\ F_i^n = \frac{OFM_{H_i} \times OFM_{W_i}}{P_i}, \\ and\ F_i^s = \frac{K_i^2 \times IFM_{Ch_i}}{S_i} \end{cases}$$

1) Variables Constraints:
   - $P_i \rightarrow \sigma_{i,p}$
   - $S_i \rightarrow \gamma_{i,p}$
   - $\boldsymbol{\varepsilon} \rightarrow z_{i,p}$

$$P_i = \sum_{p=1}^{64} \sigma_{i,p} \times x_{i,p} \quad and \quad \sum_{p=1}^{64} \sigma_{i,p} = 1$$

$$S_i = \sum_{p=1}^{64} \gamma_{i,p} \times y_{i,p} \quad and \quad \sum_{p=1}^{64} \gamma_{i,p} = 1$$

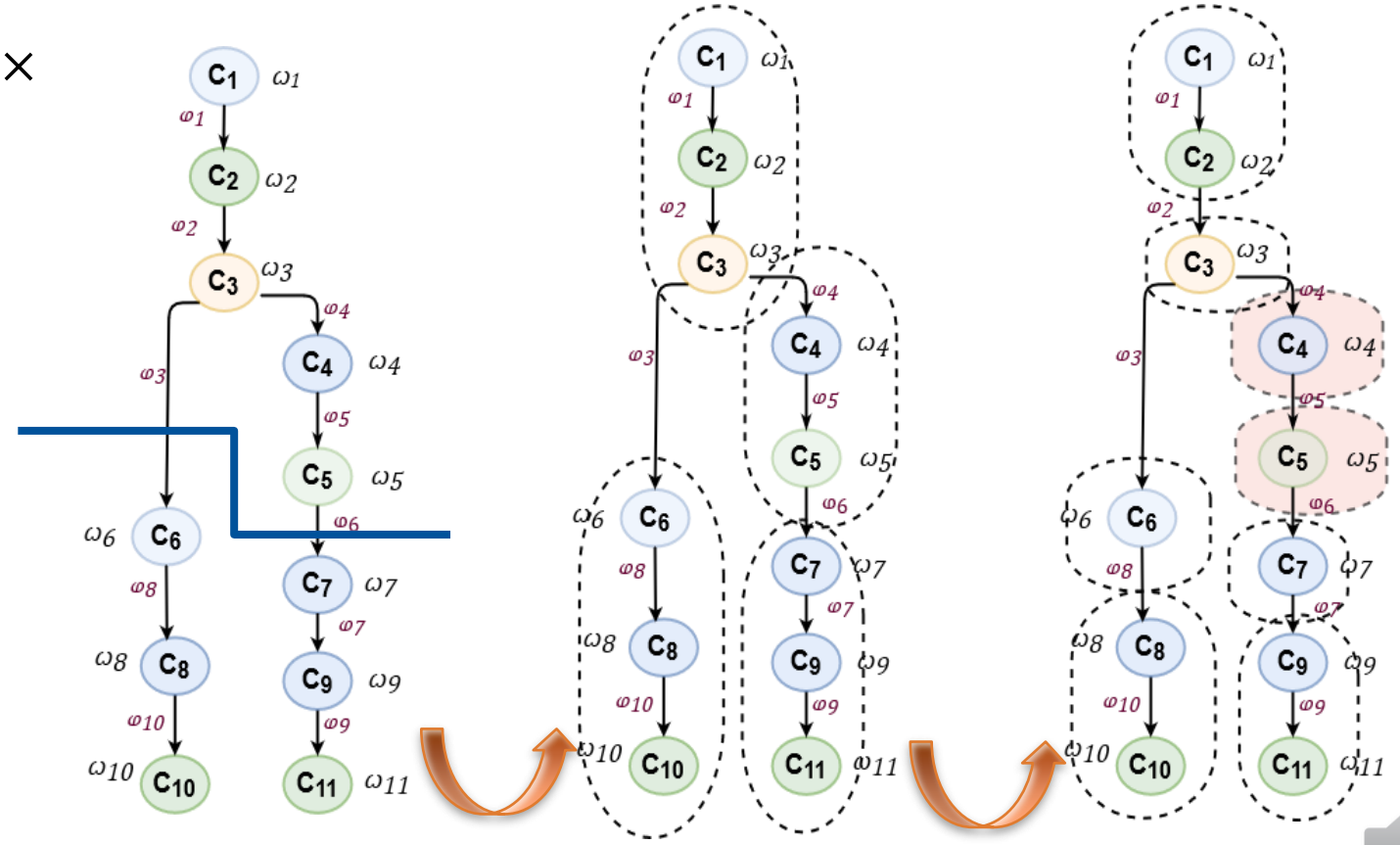
$$\epsilon_i = \sum_{p=1}^{2\times\epsilon*100} \delta_{i,p} \times z_{i,p} \quad and \quad \sum_{p=1}^{2\times\epsilon*100} \delta i, p = 1$$

1) Resources Constraints:
   - $F_{t_i}(P_i, S_i)$, a linear function that estimates the number of resources of type t demanded by the ith layer.

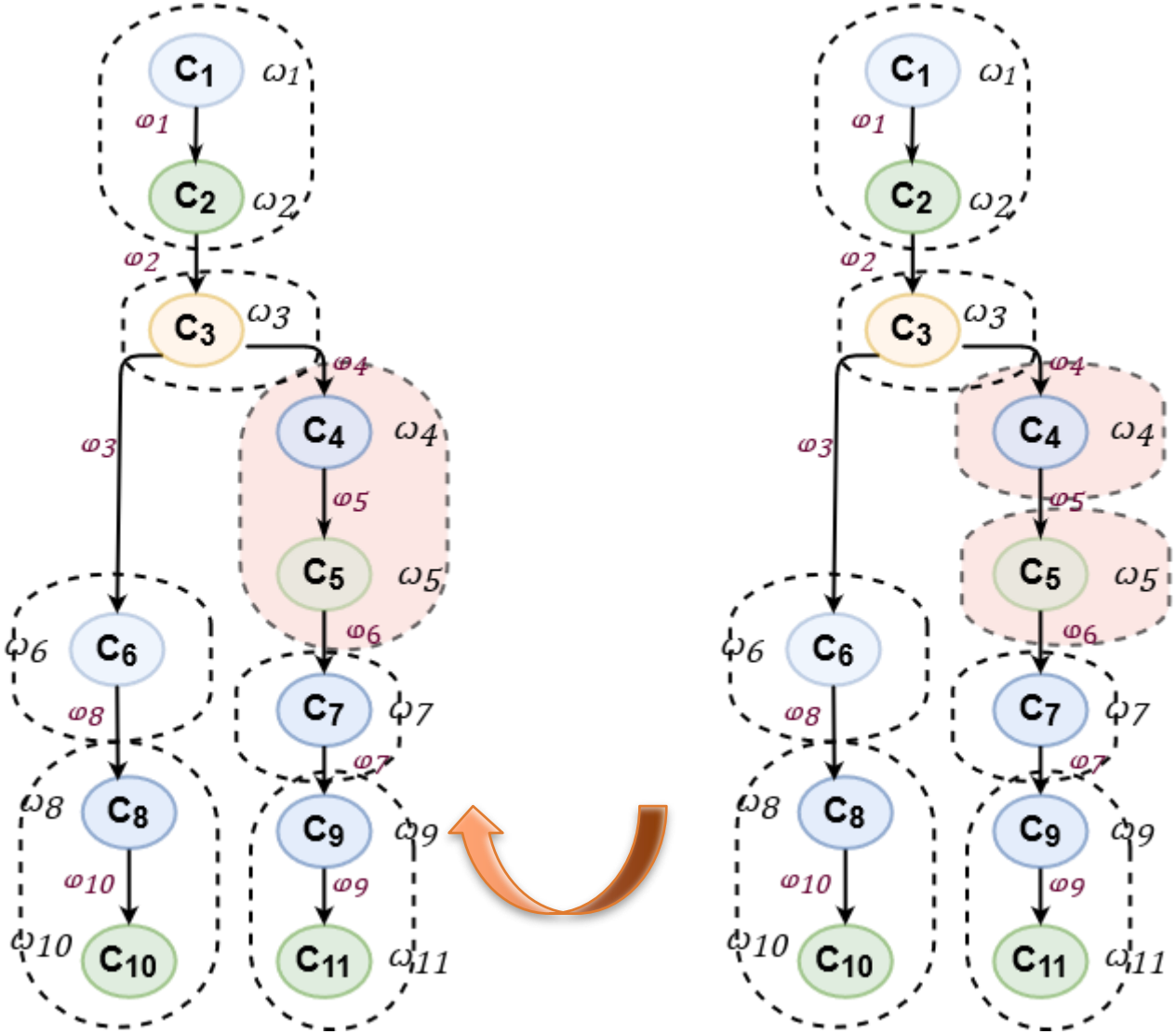
$$\begin{cases} \sum_{i=1}^{N} F_{lut_i}(P_i, S_i) \leq LUT_{VRs}, \quad \forall i = 1, ..., M \\ \sum_{i=1}^{N} F_{dsp_i}(P_i, S_i) \leq DSP_{VRs}, \quad \forall i = 1, ..., M \\ \sum_{i=1}^{N} F_{bram_i}(P_i, S_i) \leq BRAM_{VRs}, \quad \forall i = 1, ..., M \end{cases}$$

- We implement a recursive balanced bi-partitioning
- The weight of the heaviest partition ≤ $\varepsilon \times \omega(V)$
- **Stop:** k ≥ #F P Gas

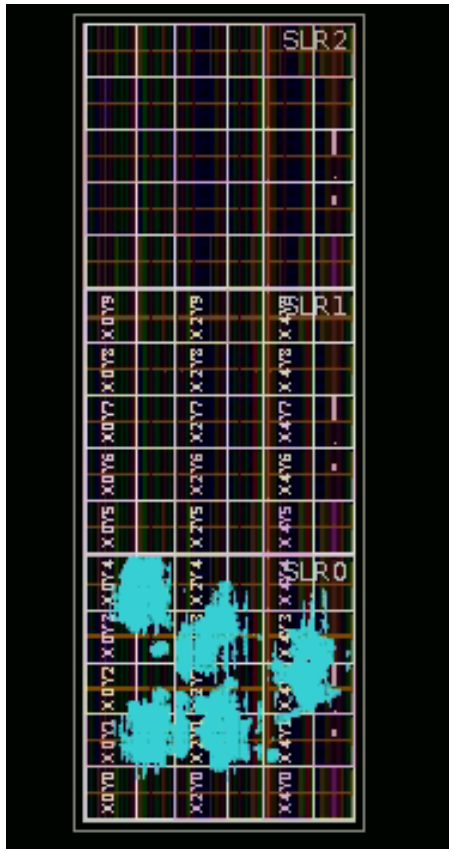
- Considering bi-partitioning produce 2^n partitions per iterations:
  - inbalanced partitions
  - or too many partitions.

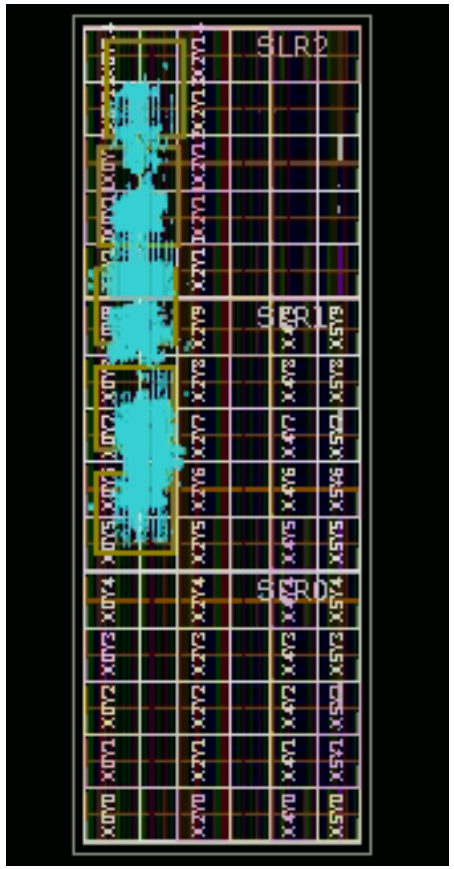


- The **refinement step** allows us:
  - to merge smaller partitions,
  - further split heavier partitions (with k ≤ #FPGAs)



Motivation Example





- Graph partitioning + Floor planning improves the frequency by 28%:
  - 325 MHz to 416 MHz.

- CAD Tools use heuristics for physical implementation
- They consider the number of cells in a design, their connections, and the target FPGA device's physical architecture to generate a circuit according to specified constraints
- Vendor tools generally achieve better QoR on smaller designs as the resource allocation problem addressed in the physical implementation is well-known to be NP-hard.
- Focusing the optimization on smaller modules (***Components***)
- lead to overall QoR improvement in a design

1 OCC

- It allows to implement and analyze (resource analysis, timing analysis, power analysis, etc) a module independently of the rest of the design.
- It enables reusing and preserving the characteristics of placed and routed modules within a top-level design.

2 Floor planning

- Utilizing pblock constraints allows carefully selecting the FPGA resources that will be used by each design component.
- Port planning with PartPins

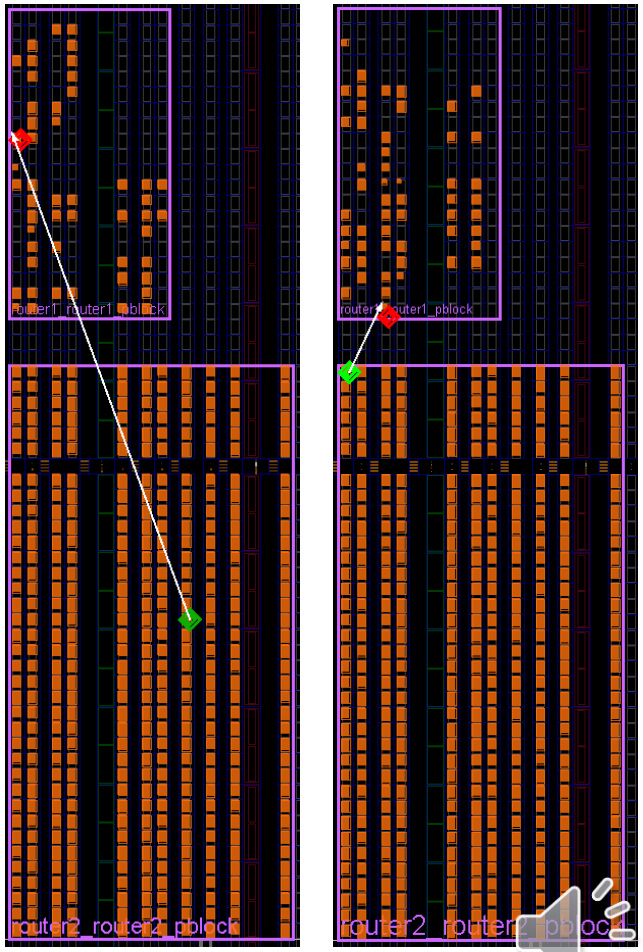
3 Clock routing

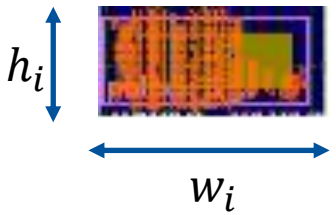
- Accurately run the timing analysis on the OOC modules

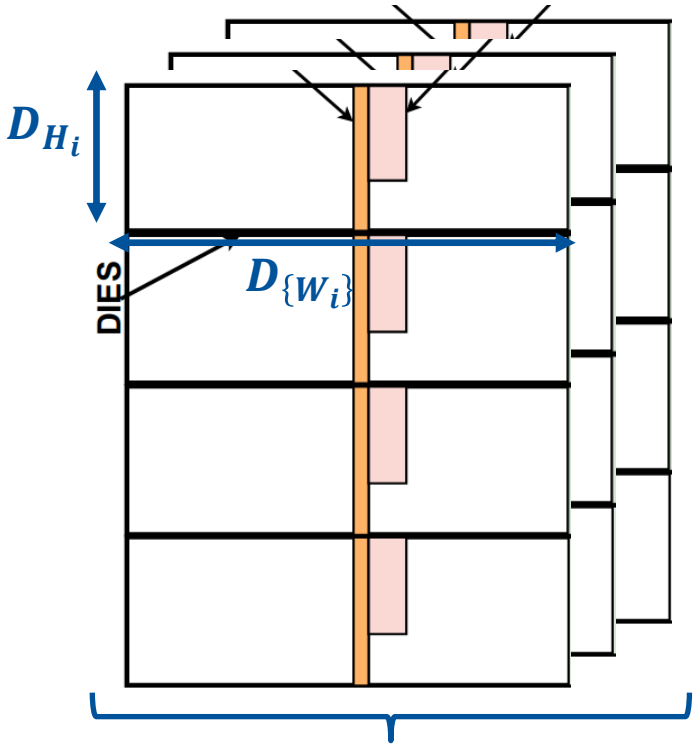
4 Lock logic

- Once a module attains a desirable performance (Fmax, area, power, etc)

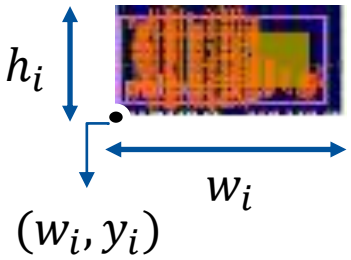
5 Checkpoint file generation

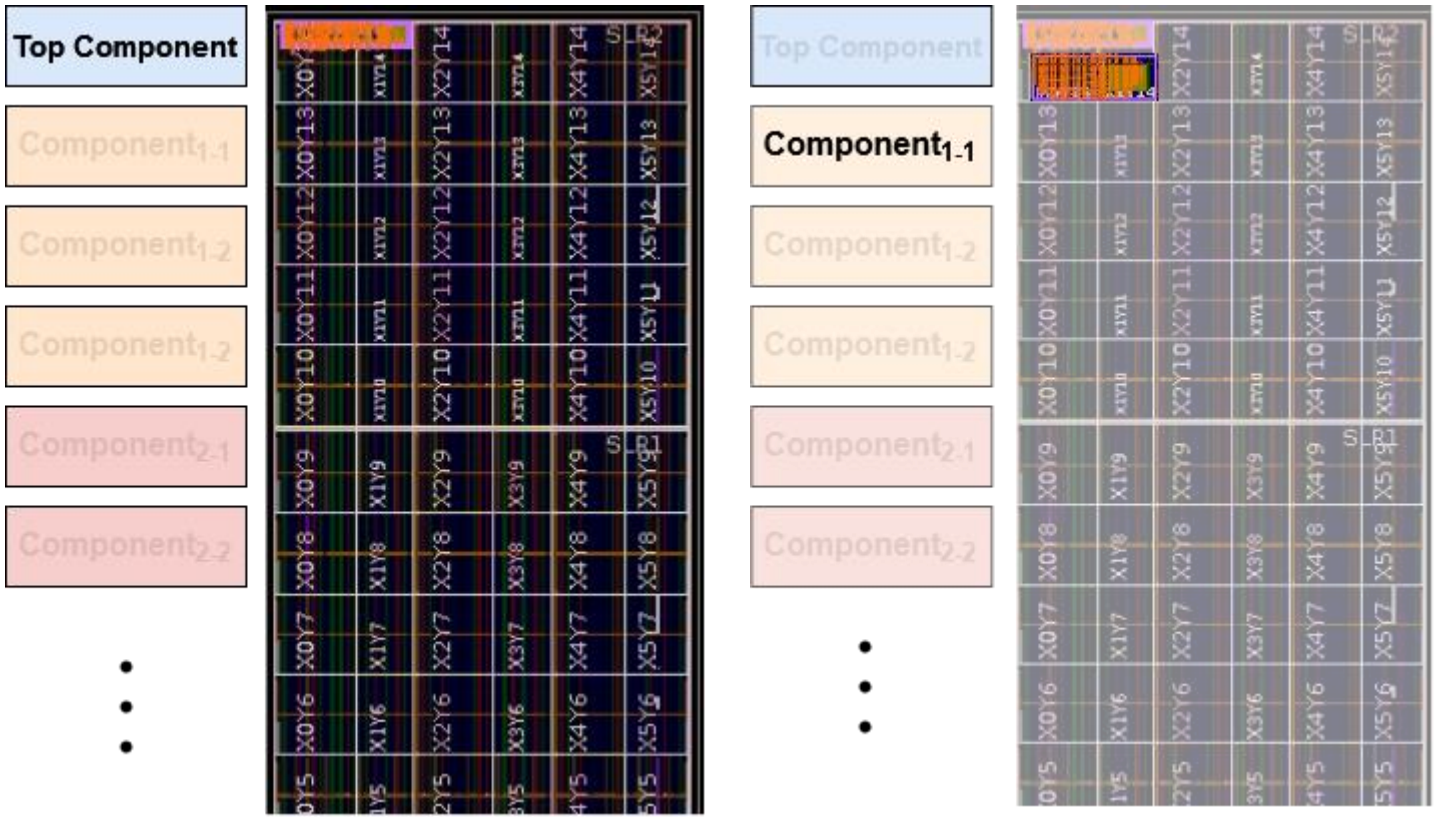






- Modules must respect linear localization constraints:
  - Two adjacent M_i and M_j are non overlapping.
  - The localization must respect placement and congestion costs.

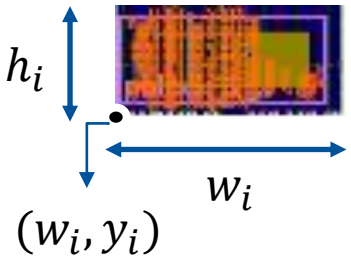


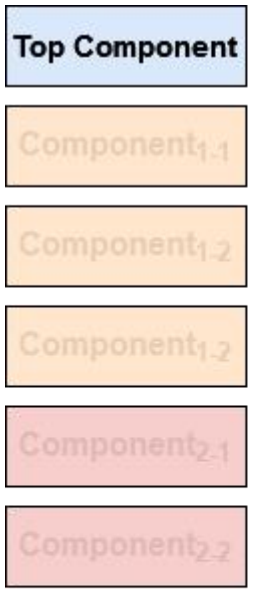


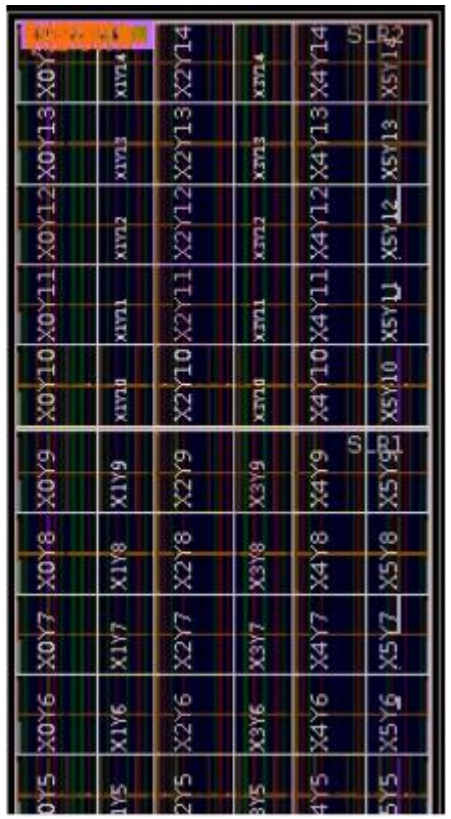
$$timing\_cost = \sum_{i=1, i<j}^{n-1} HPWL(W_{i,j})$$

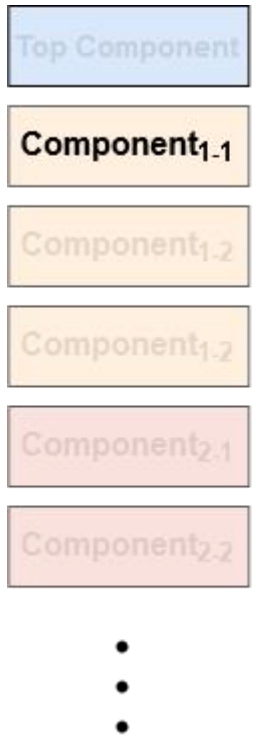
$$cgt_{coef} = \sum_{i} C_i$$

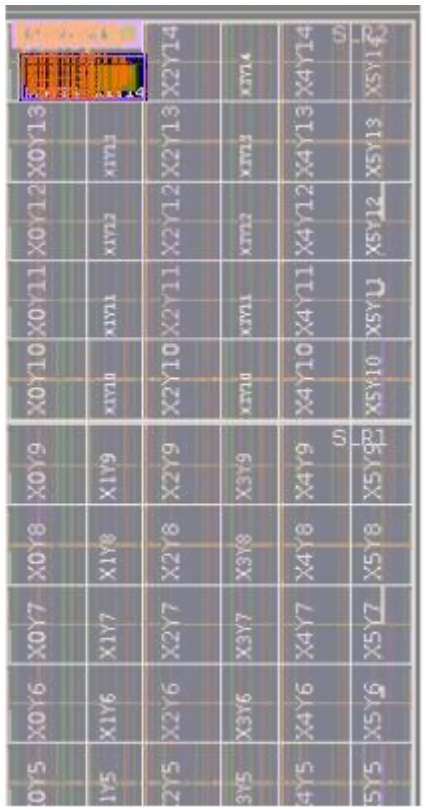
$$cgt\_cost = \sum_{W_{i,j}} \frac{\omega_{i,j} \times cgt_{coef_k}}{\#SwBox}$$

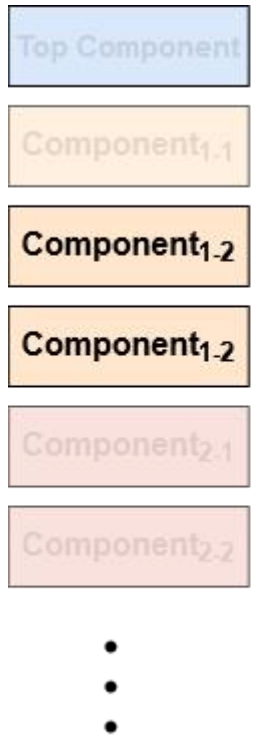

h_i
w_i
(w_i, y_i)

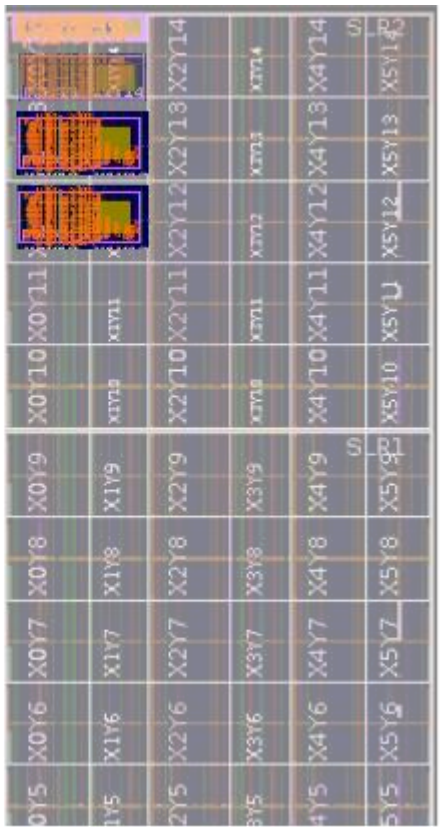

Top Component
Component1-1
Component1-2
Component1-2
Component2-1
Component2-2

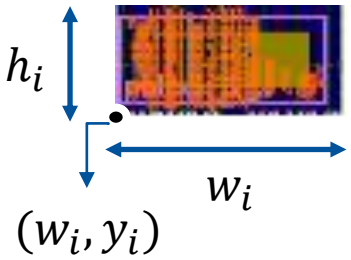


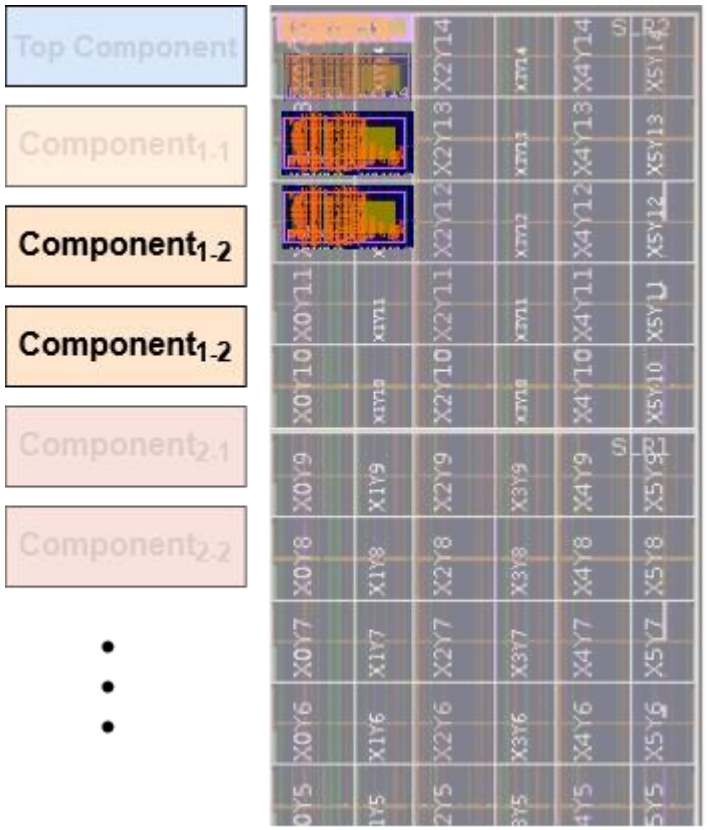

Top Component
Component1-1
Component1-2
Component1-2
Component2-1
Component2-2

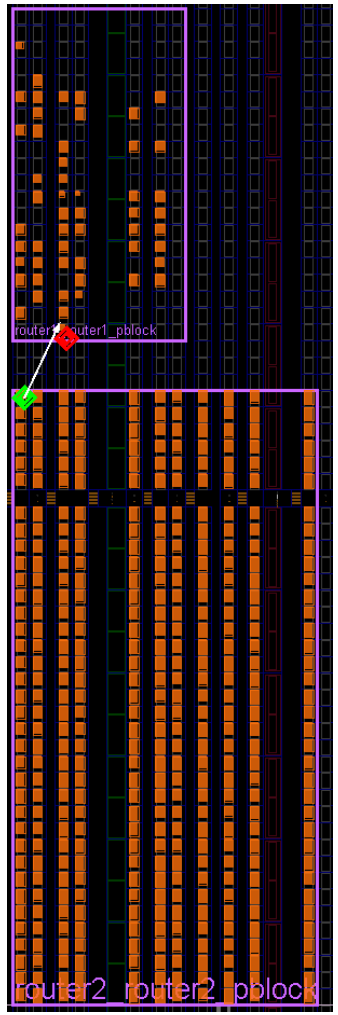



Top Component
Component1-1
Component1-2
Component1-2
Component2-1
Component2-2





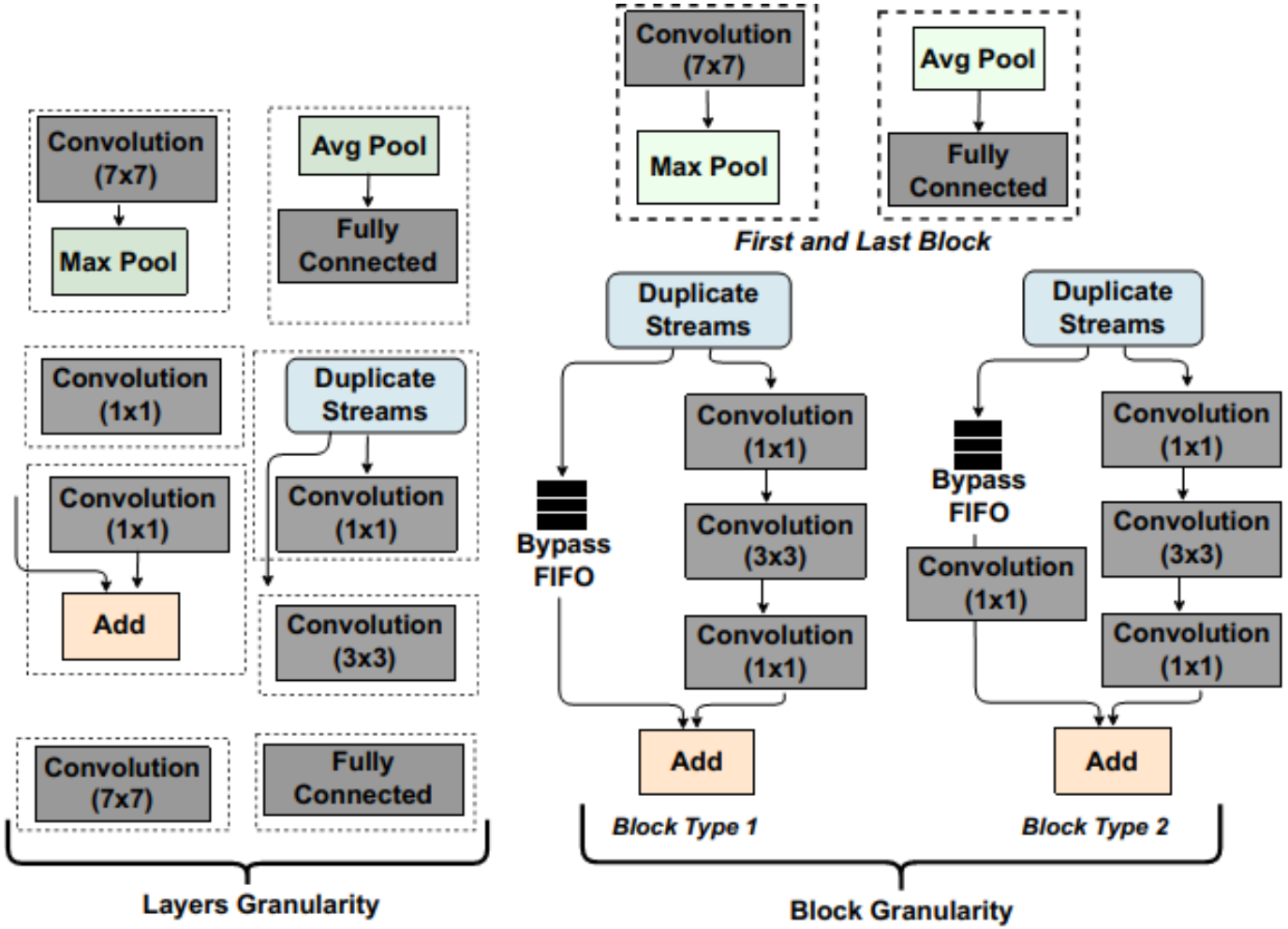




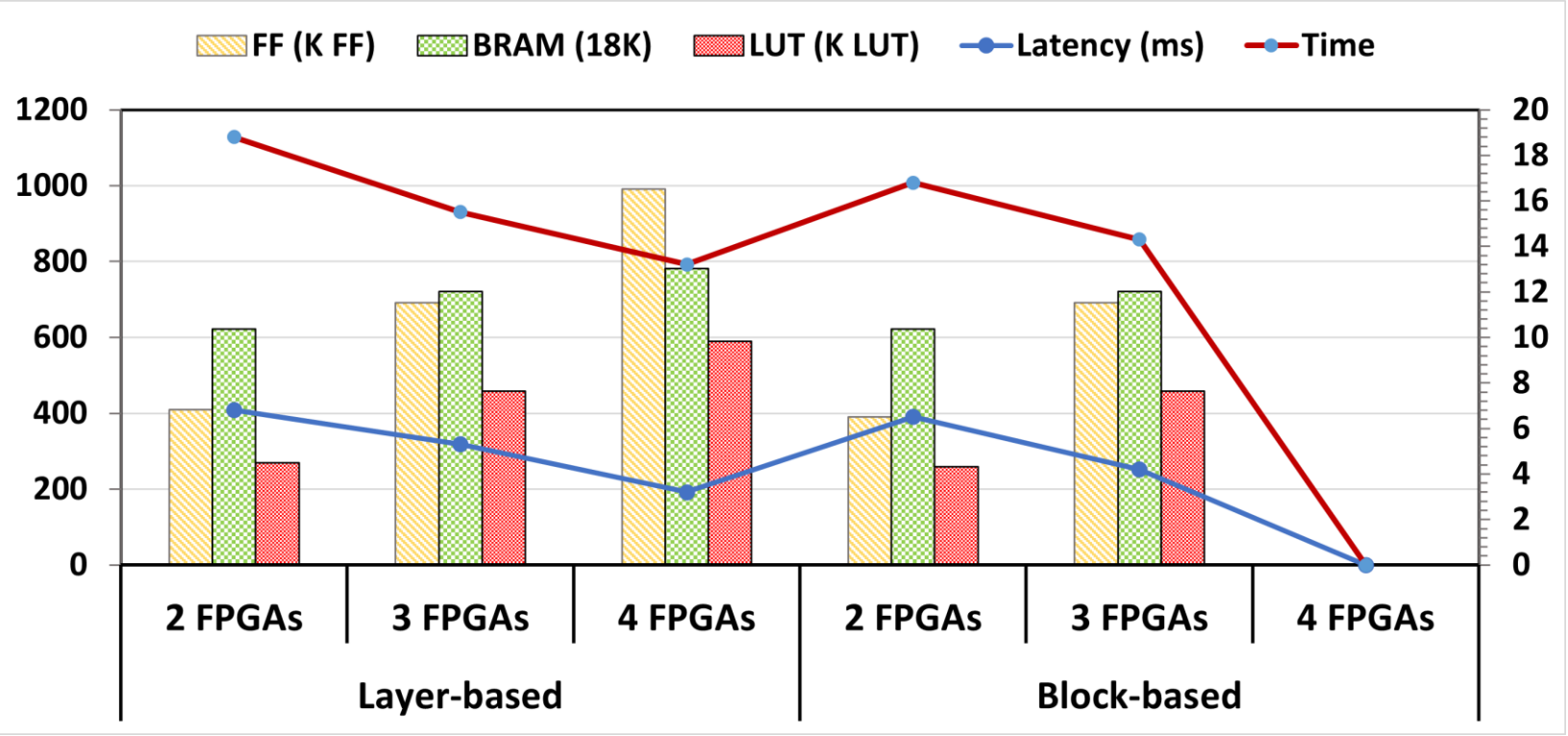
1. Create inter-connected nets between components
2. Inter-component routing

- Experiment has been conducted on ResNet-50
- The hardware is generated using Vivado v2020.2 and RapidWright v2020.
- The components are implemented with vivado HLS.
- The MIP is solved with LocalSolver.

- Granularity Exploration:



- Granularity Exploration:



	Layer-based ResNet			Block-based ResNet			Baseline ResNet		
	KFF	KLUTs	BRAM	KFF	KLUTs	BRAM	KFF	KLUTs	BRAM
Resources	741 ((↓ 26%)	421 (↓ 24%)	821.5 (≅)	801 (↓ 16%)	479 (↓ 10%)	761.6 (↓ 7%)	935	526	822
Latency (ms)	4.8 (↓ 31%)			4.2 (↓ 33 %)			6.3		
Frequency (MHz)	276 (↑ 37%)			252 (↑ 25.3%)			201		
Avg. Power (W)	208			225			235		
Energy Efficiently	29.79			27.98			22.15		

	Layer Granularity		**ResNet**	
	Custom API	Inter-node Routing	Synthesis	P&R
Time (hours)	1.14	4.82	4.16	8.9
Ratio	∼ 17.4%	82.6%	35.6%	64.4%
Total (hours)	**5.96** (2.18× ↓)		**13.02**	

	Block Ganularity	
	Custom API	Inter-node Routing
Time (hours)	0.32	3.63
Ratio	8.10%	91.8%
Total (hours)	**3.95** (3.21× ↓)	

Conclusion

- We propose a framework to accelerate model inference on a multi-die Cloud FPGA Platform.
- It takes as input the model definition and performs an intensive search in the form of a MIP problem to determine each layer's highest degree of parallelism considering the platform constraints.
- The graph is then partitioned, and the resulting sub-graphs are allocated to the FPGAs' dies.
- Experiments and results show that our approach improves latency and maximum frequency, with little to no impact on the number of resources used.
- Future works we intend to extend to deeper neural networks.

UF Herbert Wertheim College of Engineering
UNIVERSITY of FLORIDA

THANK YOU