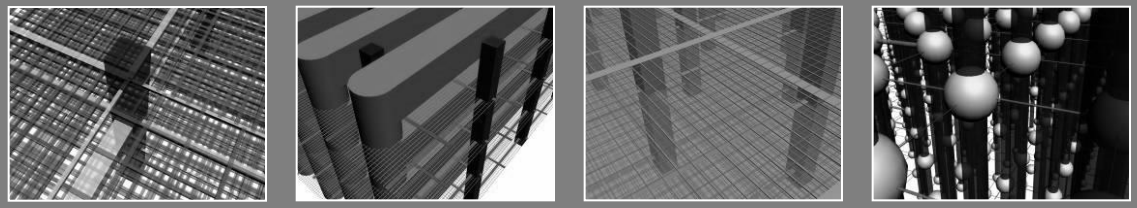
Monolithic 3D IC Design for Deep Neural Networks

with Application on Low-power Speech Recognition



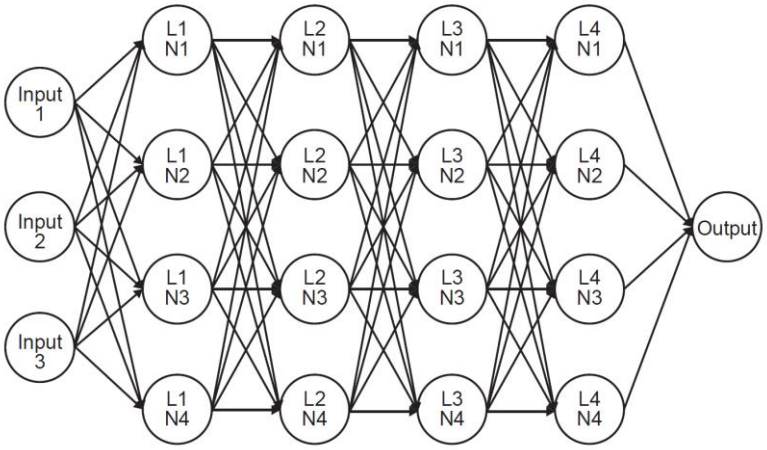
Kyungwook Chang[1], Deepak Kadetotad[2], Yu (Kevin) Cao[2], Jae-sun Seo[2], and Sung Kyu Lim[1]

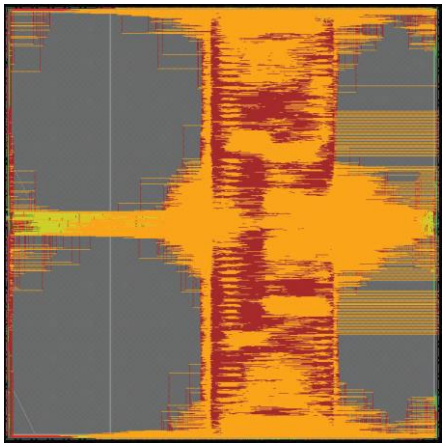
[1] School of ECE, Georgia Institute of Technology
[2] School of ECEE, Arizona State University

Agenda

- Introduction
- Deep Neural Network for Speech Recognition
- Full-Chip Monolithic 3D IC Design Flow
- DNN Architecture Description
- Circuit Design Discussions
- Architectural Impact Discussions
- Conclusion

DNN Hardware Design



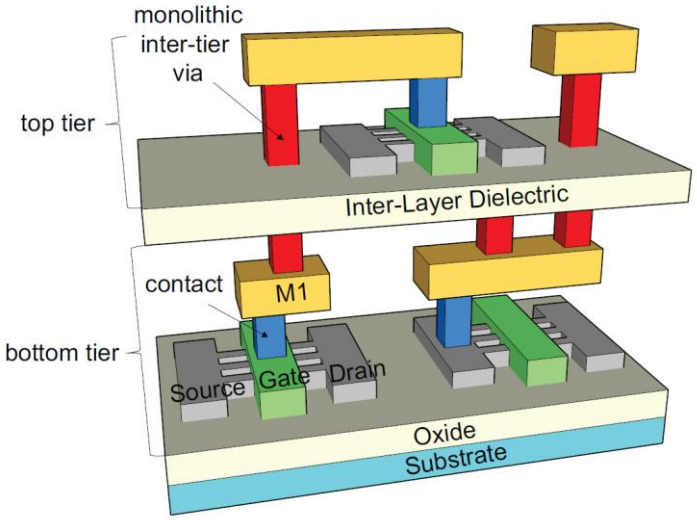


- **Challenges**
  - The computations require large amount of energy
  - Excessive memory is needed to store the weights
  - Prohibitive wire overhead due to a large number of connections

Optimization for both architecture and physical implementation is required

Monolithic 3D IC (M3D)

- **Placing cells in 3D space**
  - **Sequentially fabricating transistors on multiple tiers**

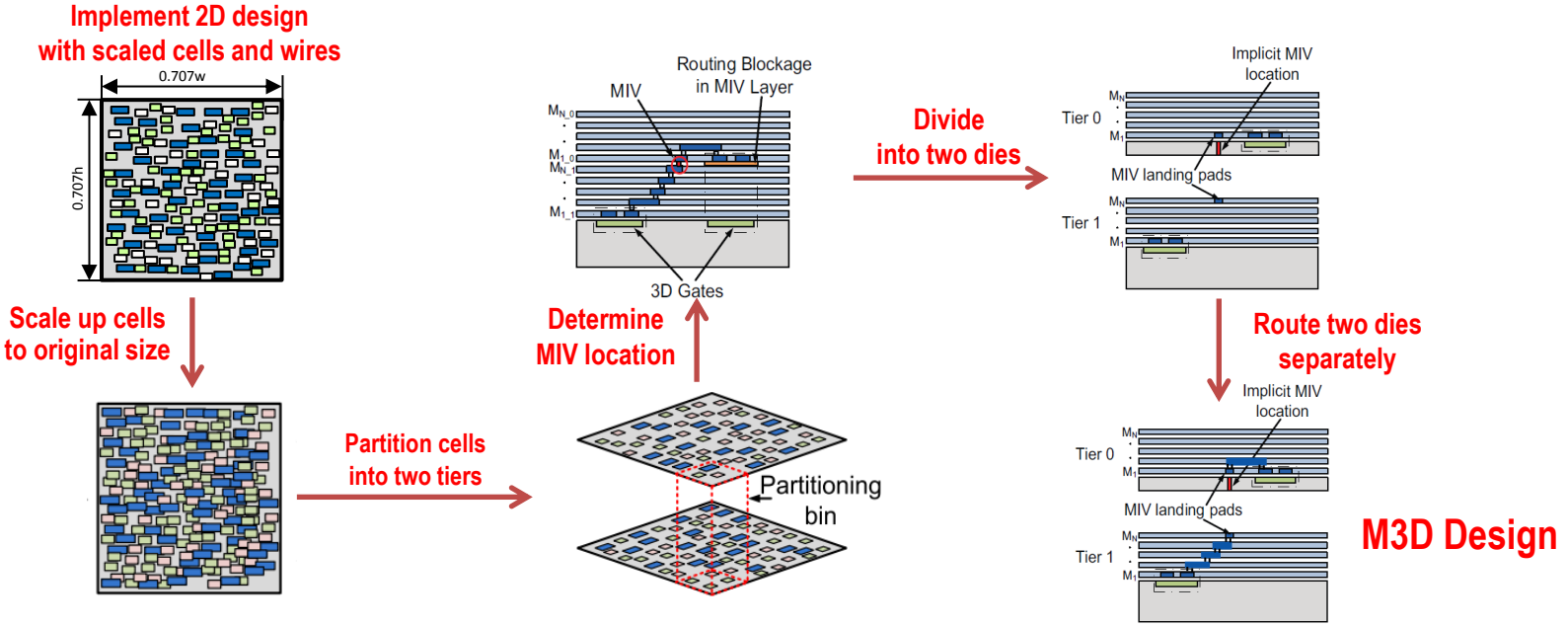


- **Utilizing Monolithic Inter-tier Vias (MIVs) to connect cells on different tiers. Compared to Through Silicon Vias (TSV)**
  - **Size: MIV << TSV → Achieve higher MIV density**
  - **RC: MIV << TSV → Achieve higher performance**

M3D Implementation Methodology

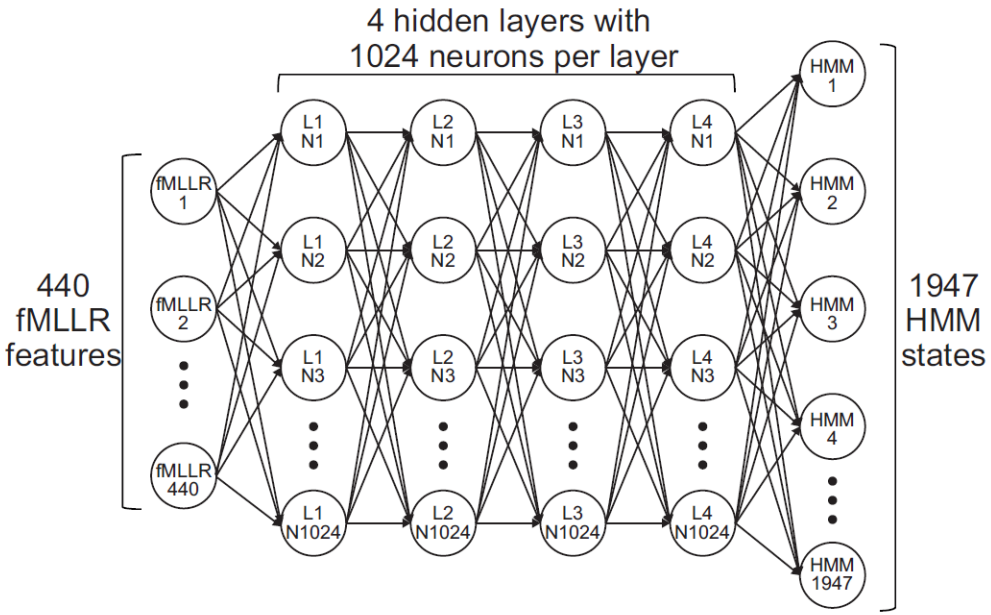
- **Optimization on physical implementation**
- **No EDA tools supporting 3D IC implementation**

  Use tricks to place cells in 3D space with 2D tools
- **Shrunk2D M3D implementation flow**



Shreepad Panth, Kambiz Samadi, Yang Du, and Sung Kyu Lim, "Design and CAD Methodologies for Low Power Gate-level Monolithic 3D ICs," IEEE International Symposium on Low Power Electronics and Design, 2014

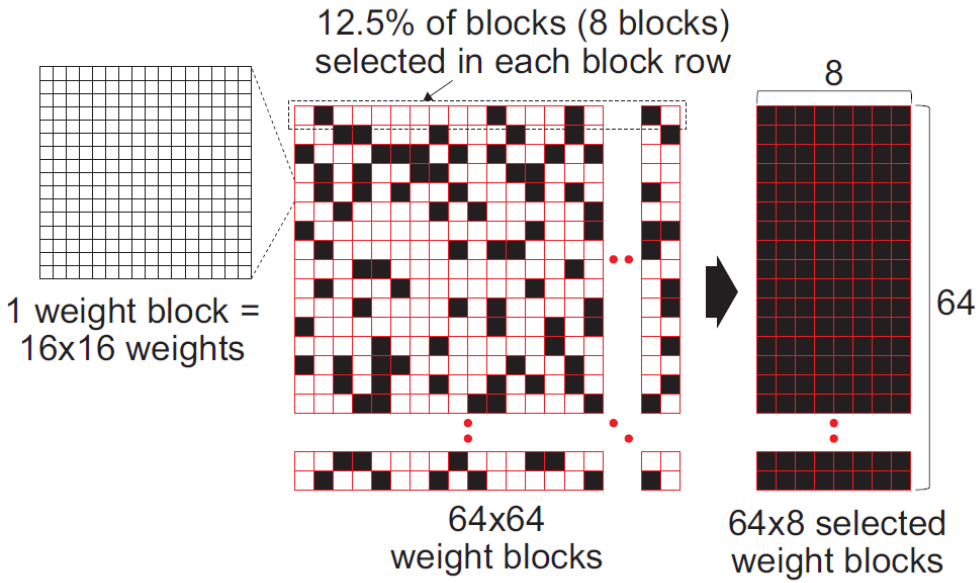
DNN for Speech Recognition



- **Training**
  - **Objective function:** $E = -\sum_{i=1}^{N} t_i \cdot \ln(y_i)$
  - **Weight update function:** $(W_{ij})_{k+1} = (W_{ij})_k + C_{ij}\left[-lr(\Delta W_{ij})_k + m(\Delta W_{ij})_{k-1}\right]$
- **Classification**
  - **Feed-forward computation**
  - **Matrix-vector multiplication of weight matrices and neuron vectors**

Coarse-Grain Sparsification (CGS)

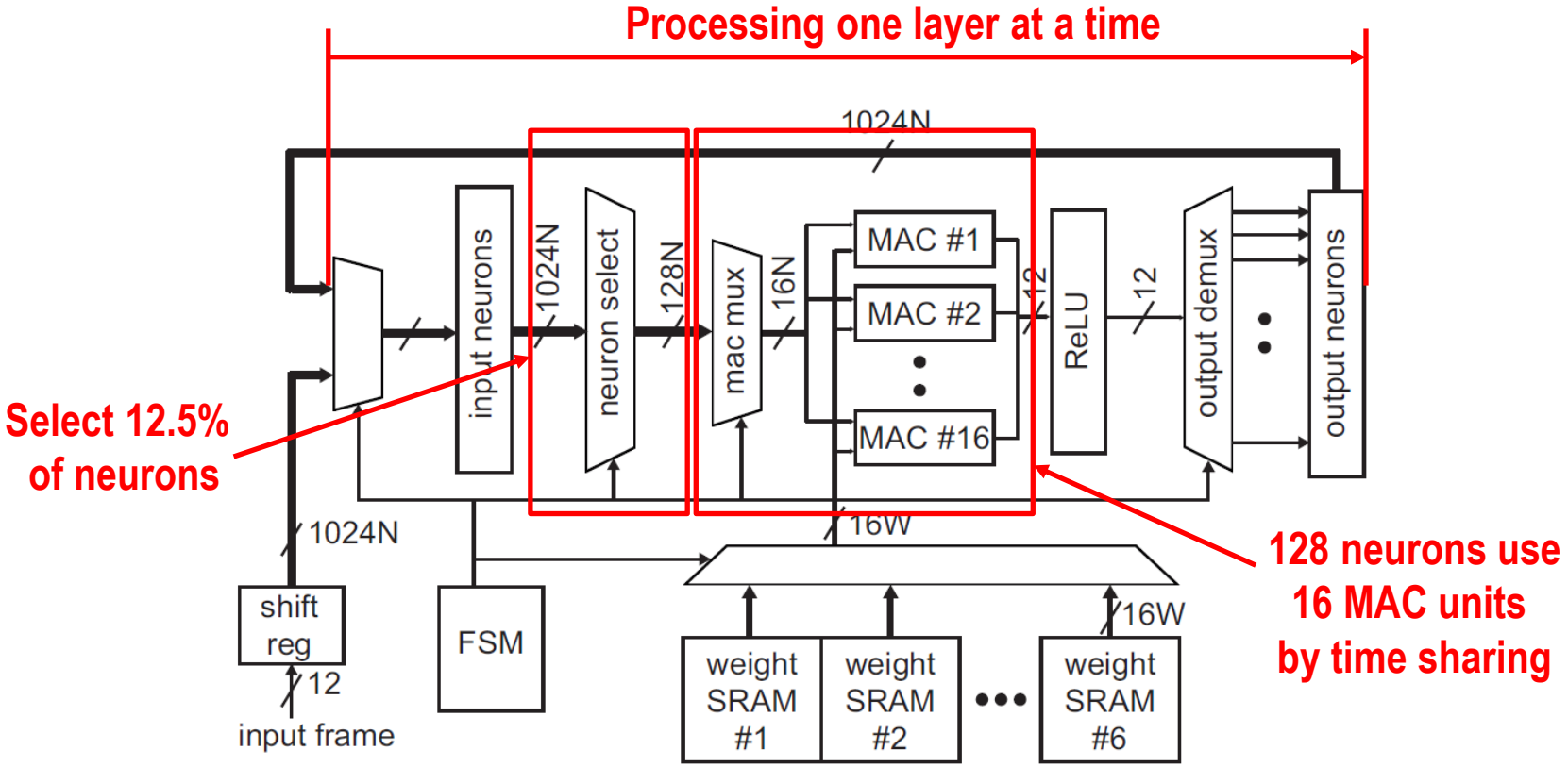
- **Architectural optimization for hardware implementation by Memory compression**
  - **Connections between two consecutive layers are compressed in block-wise manner**



D. Kadetotad et al., "Efficient Memory Compression in Deep Neural Networks using Coarse-Grain Sparsification for Speech Applications," IEEE International Conference on Computer-Aided Design, 2016

DNN Architecture Description

- Operating one layer at a time – requires multiple iterations
- Target compression rate of CGS – 87.5%



DNN Architecture Description

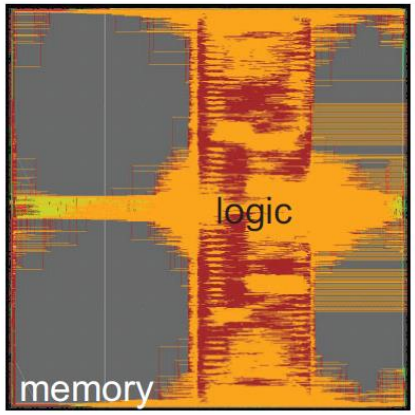
- **DNN with CGS block size of 16x16 (DNN CGS-16) and 64x64 (DNN CGS-64) are selected for the experiment of this paper**

parameters	DNN CGS-16	DNN CGS-64
block size	16x16	64x64
compression rate	87.5%	87.5%
size of coefficient register file	15,360 bits	640 bits
size of SRAM for weights	6Mb	6Mb
phoneme error rate	**19.8%**	**19.8%**

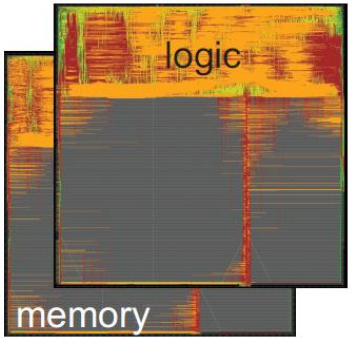
Experimental Setup

- Designs: DNN CGS-16 and DNN CGS-64
- Technology: TSMC 28nm (CLN28HPM)
- Clock frequency: 400MHz
- Initial standard cell density: 65%
- Memory floorplans for M3D design
  - M3D-both: Memory blocks exist on both tiers
  - M3D-one: Memory blocks are placed on a single tier (bottom tier)

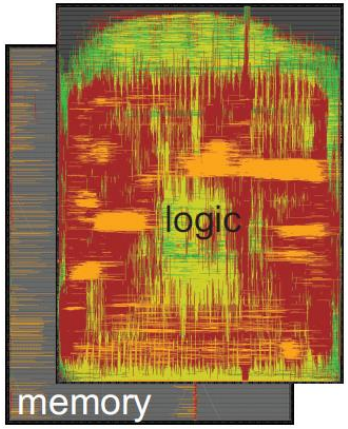
Die Shots



2D

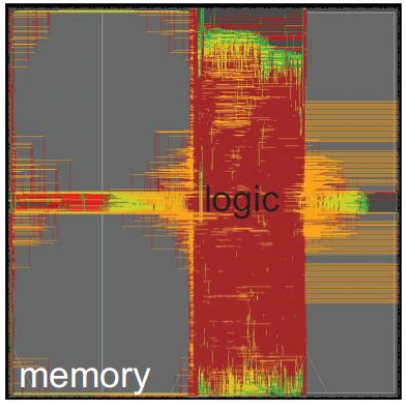


M3D-both

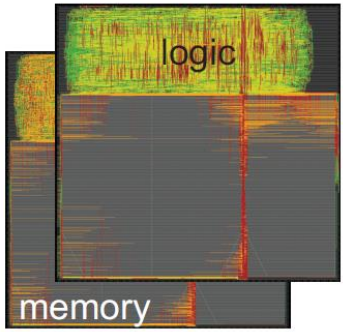


M3D-one

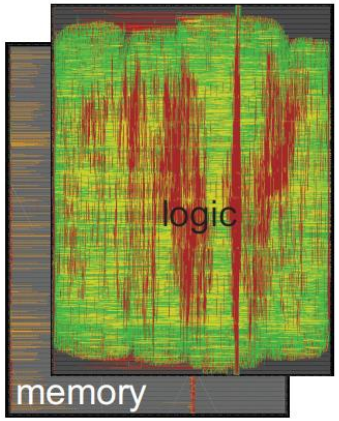
DNN CGS-16



2D



M3D-both



M3D-one

DNN CGS-64

M3D-both vs. M3D-one

- **Examine the impact of memory floorplan**

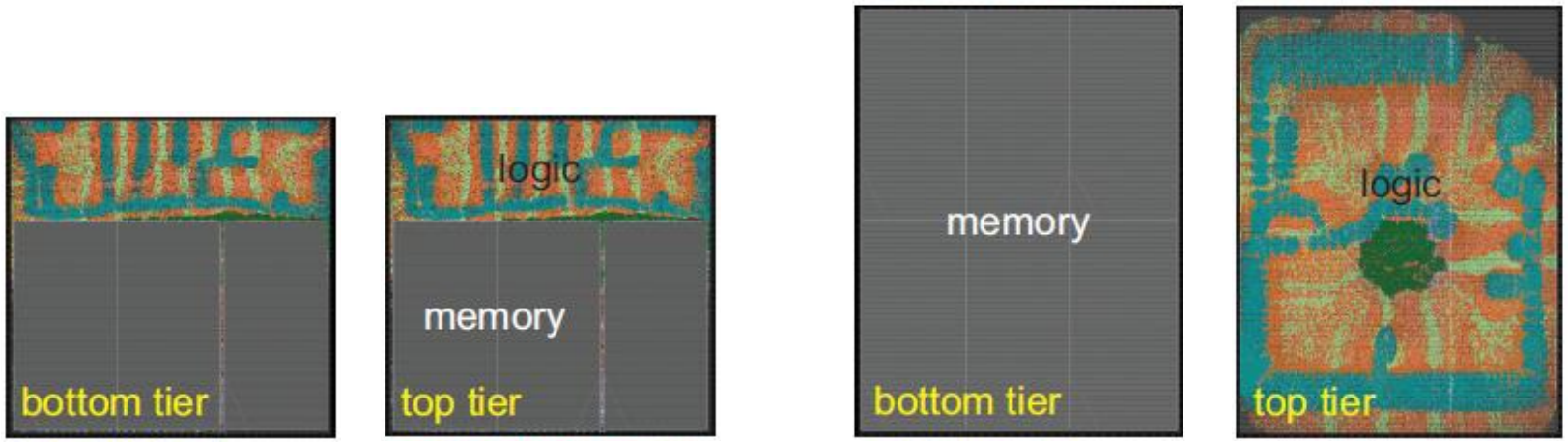
	2D	CGS-16 M3D-both		CGS-16 M3D-one	
footprint (um)	1411x1411	1010x984	-50.1%	996x1322	-33.9%
cell area (mm^2)	0.505	0.431	-14.6%	0.511	1.1%
wirelength (m)	12.089	8.469	-29.9%	12.225	1.1%
MIV count	-	77,536		1,776	
pin cap (pF)	943.3	788.0	-16.5%	1,004.1	6.4%
wire cap (pF)	2,216.8	1,440.8	-35.0%	2,087.4	-5.8%
total cap (pF)	3,160.1	2,228.7	-29.5%	3,091.6	-2.2%

	2D	CGS-16 M3D-both		CGS-16 M3D-one	
internal power (mW)	91.3	76.7	-16.0%	90.3	-1.1%
switching power (mW)	48.6	31.6	-35.0%	46.5	-4.3%
leakage power (mW)	1.3	1.2	-6.6%	1.3	0.5%
total power (mW)	141.1	109.6	-22.3%	138.0	-2.2%

M3D-both shows better performance in all aspects

M3D-both vs. M3D-one (cont'd)

- **Footprint of M3D-one is determined by memory blocks**
  - **Since "area of memory > area of logics", footprint reduction of M3D-one is smaller than M3D-both**
- **In M3D-one, logic gates are spread out across the top tier, we increases wire-length**



M3D-both M3D-one

Footprint management and tier partitioning is important when there are large memory modules

M3D-both vs. M3D-one (cont'd)

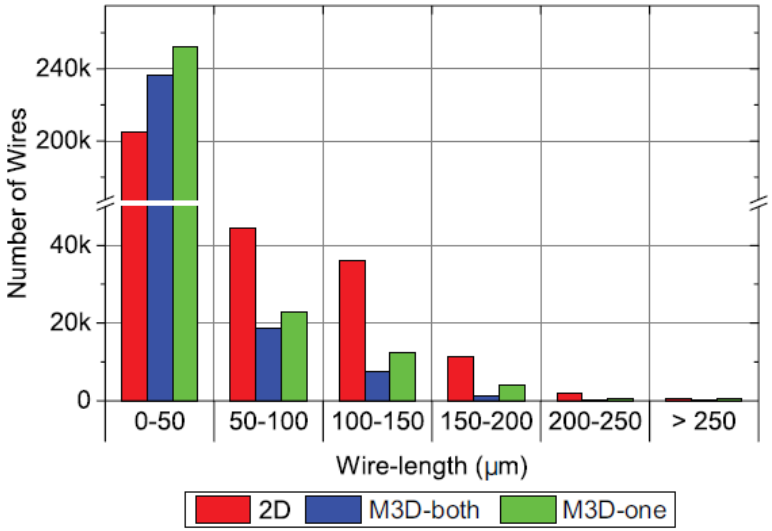
- **Dynamic power breakdown**

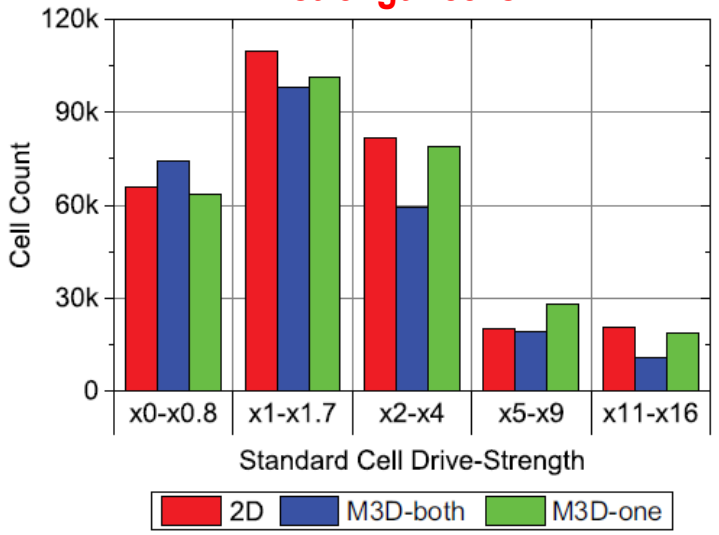
$$P_{dyn} = P_{sw} + P_{int}$$
$$= P_{sw,wire} + P_{sw,pin} + P_{int}$$
$$= \alpha_{out}(C_{wire} + C_{pin})V_{DD}{}^2 f_{clk} + P_{int}$$

Depends on wire-length

Depends on standard cell area

Standard cell area also depends on wire-length since longer wire-length requires higher drive-strength cells





Less power saving of M3D-one is attributed to the less footprint reduction

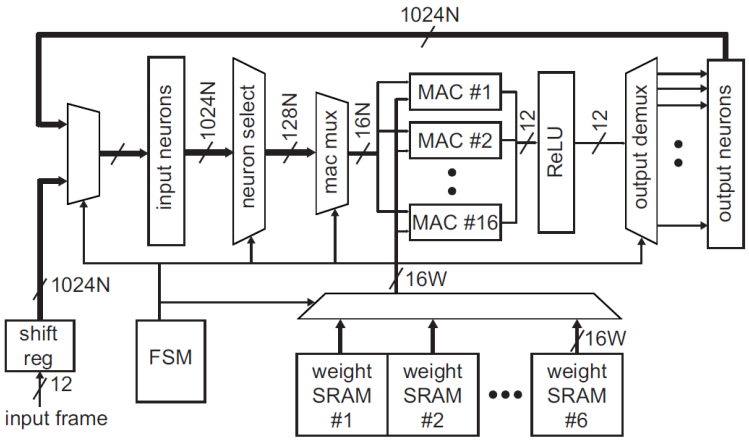
CGS-16 vs. CGS-64

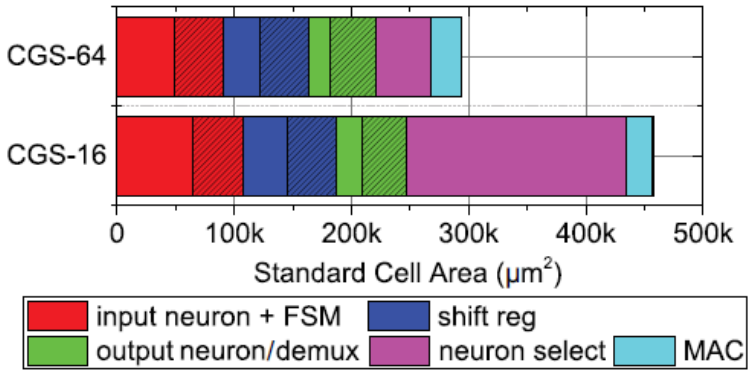
- **Examine the impact of DNN architecture**

	CGS-16 2D	CGS-16 M3D-both		CGS-64 2D	CGS-64 M3D-both	
footprint (um)	1411x1411	1010x984	-50.1%	1411x1411	1010x984	-50.1%
cell area (mm^2)	0.505	0.431	-14.6%	0.314	0.269	-14.3%
wirelength (m)	12.089	8.469	-29.9%	5.631	3.734	-33.7%
MIV count	-	77,536		-	48,636	
pin cap (pF)	943.3	788.0	-16.5%	520.8	390.8	-25.0%
wire cap (pF)	2,216.8	1,440.8	-35.0%	920.1	573.7	-37.7%
total cap (pF)	3,160.1	2,228.7	-29.5%	1,440.9	964.4	-33.1%

	CGS-16 2D	CGS-16 M3D-both		CGS-64 2D	CGS-64 M3D-both	
internal power (mW)	91.3	76.7	-16.0%	86.8	76.1	-12.3%
switching power (mW)	48.6	31.6	-35.0%	41.2	30.2	-26.7%
leakage power (mW)	1.3	1.2	-6.6%	1.1	1.1	-4.7%
total power (mW)	141.1	109.6	-22.3%	129.1	107.3	-16.9%

CGS-16 vs. CGS-64 (cont'd)





Non-dashed: Combinational logics
Dashed: Sequential logics

- **Combinational logic occupies more portion in CGS-16**
  - **Due to more complex 'neuron selection' logic of CGS-16**
- **The number of sequential logic does not benefit from M3D**

Design with large portion of combinational logics benefits more from M3D

Feed-Forward Classification vs. Pseudo-Training

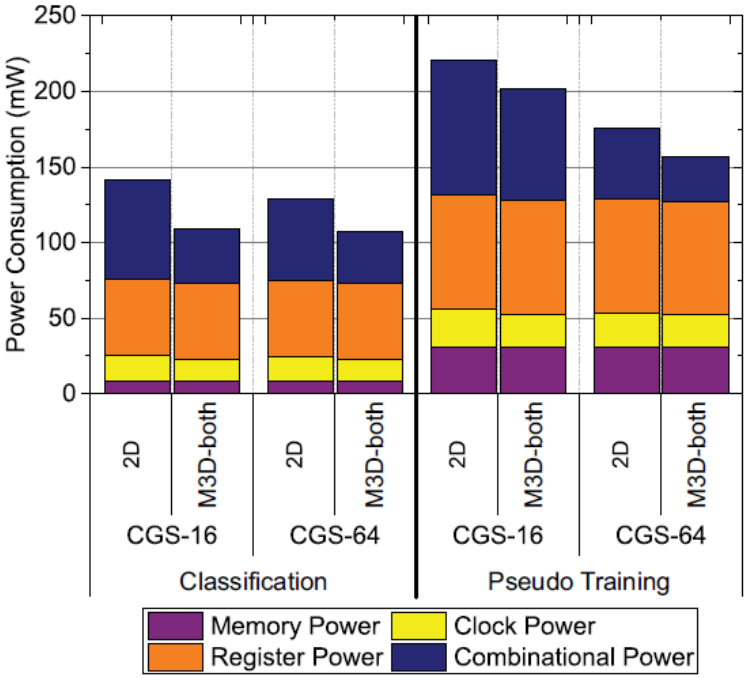
- Examine the impact of workloads
- Feed-forward classification
- Training: Current architecture supports only offline training
  - Created customized test vector
  - Pseudo-training: Feed-forward classification + weight write phases

	Feed-forward classification			Pseudo-training		
	2D	M3D-both		2D	M3D-both	
internal power (mW)	91.3	76.7	-16.0%	150.4	142.8	-5.1%
switching power (mW)	48.6	31.6	-35.0%	68.4	57.1	-16.6%
leakage power (mW)	1.3	1.2	-6.6%	1.3	1.2	-6.8%
total power (mW)	141.1	109.6	-22.3%	220.0	201.0	-8.6%

M3D shows more benefit in feed-forward classification workloads

Feed-Forward Classification vs. Pseudo-Training (cont'd)

- **Pseudo-training**
  - **Feed-forward classification + weight write phases**
  - **Involves more memory operations**
  - **More power consumption on sequential logics, which cannot be reduced effectively with M3D**



M3D shows bigger impact on the compute-intensive workloads

Conclusion

- M3D effectively reduces the total power consumption of deep neural network hardware by reducing wirelength and standard cell area
- DNN with large amount of memory requires memory block partitioning to maximize total power saving
- M3D shows larger power savings with smaller CGS block sizes mainly since it consist of more combinational logics
- Compute-intensive classification workload offers more power saving than memory-intensive training workload. This may not be true for other DNN architectures, but analysis method used in this paper is useful to study practical ASIC implementation of DNN