A 3DIC system to aid in the acceleration of systems that employ multiple instances of artificial neural networks

Final exam : Lee B. Baker

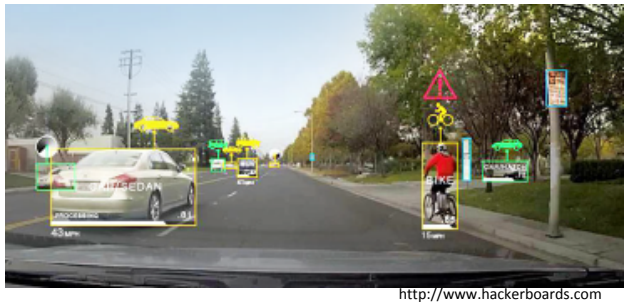
Advisor: Paul Franzon

Introduction

- Recently Artificial Neural networks (NN) have demonstrated superior performance in classification and function approximation

Deep Neural Networks (DNN) have been very successful in image processing applications [Kri12]

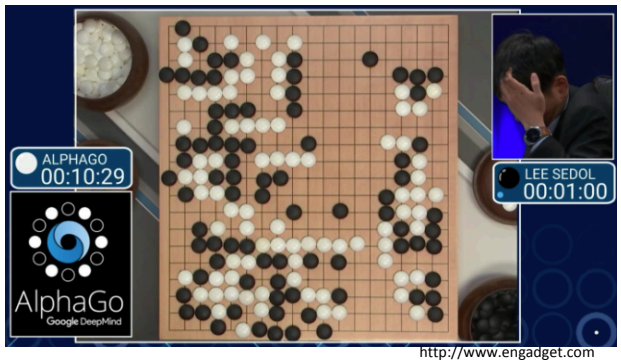
– it is anticipated that DNNs will be employed more and more in self-driving cars


http://www.hackerboards.com

http://www.BGR.com

Reinforcement learning is gaining popularity

– alphaGo employed reinforcement learning with deep neural networks [Mad14]



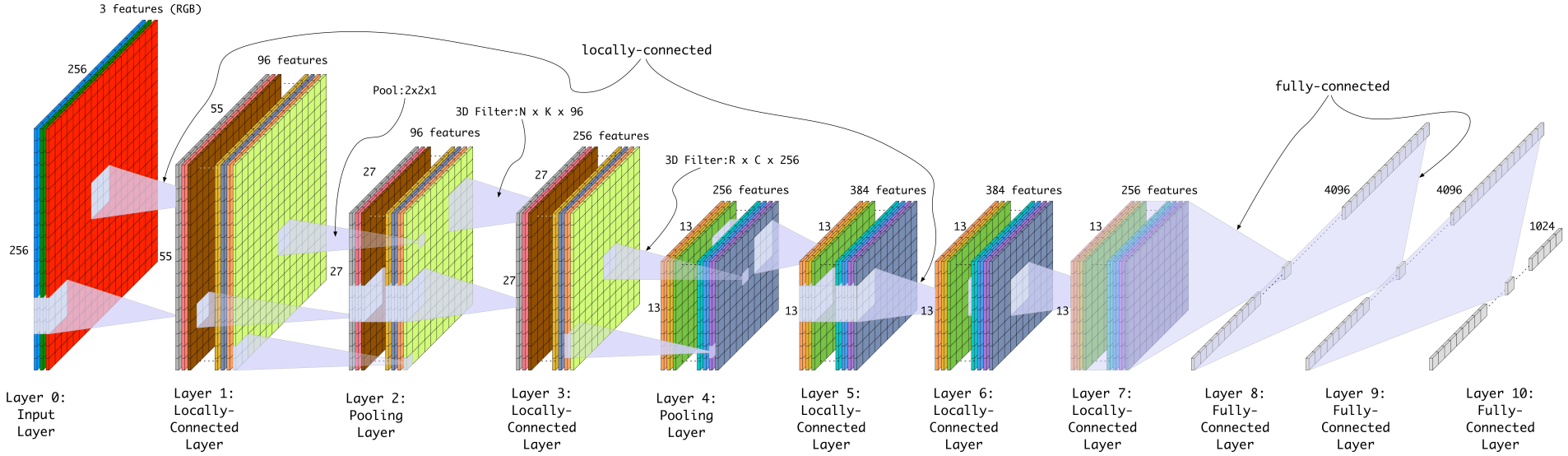
http://www.engadget.com

Introduction

- What about applications that will and do require multiple NN processors running at or near real-time?
  - drone scanning in many directions during navigation
    - looking down different than looking forward??
  - aircraft performing system diagnostics during flight
    - observing thermal profiles
    - observing vibrations
  - airport security
    - face recognition
    - body thermal profiles
    - motion

Introduction

- Most useful ANNs are big
  - 100's of thousands of artificial neurons
  - real-time processing requires lots of memory bandwidth and storage



Research

- Most research focused on CNNs
  - can take advantage of SRAM using parameter reuse
- A lot of focus on server applications
  - perhaps lots of research $$ available??
  - server applications can take advantage of SRAM for batch processing
- But many DNNs are fully-connected [XX]
  - LSTM and MLPs
  - classifier stage of all ANNs including CNN

Mission Statement

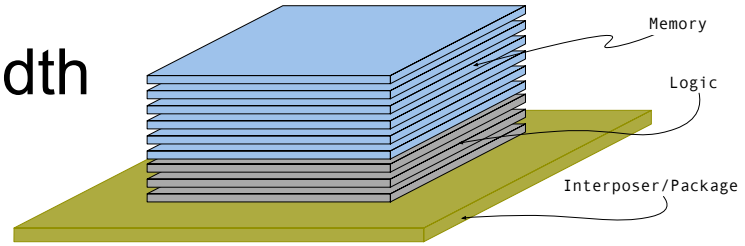
- Real world applications will require multiple disparate NNs to be solved simultaneously
  - If ANNs fulfill their potential, we believe that systems employing ANNs will utilize them for various functions, such as engine monitoring, anomaly detection, navigation etc. all within the same system
- We need the capacity provided by Dynamic Random Access Memory (DRAM)
  - SRAM is easy to use, but capacity limitations impose ANN size restrictions. DRAMs are difficult to use but they provide the capacity we need.
  - Much of the research has employed SRAM. Perhaps because it provides high bandwidth but could it also be because its easy to use
  - Some research has employed DRAM, but its often used as a feeder to an SRAM based implementation.
- Need to consider the system impact
  - Research often focuses on point problems. This isn't unreasonable, research institutions have limited resources.

Problem

- Many embedded applications will/do have power, space and weight limitations
  - to achieve near or at real-time processing, current solutions require high power and high real-estate
    - GPU solutions have large real-estate and power requirements (~100-200W)
    - ASIC's demonstrate significant improvements over GPUs
    - Most have memory bandwidth limitations when reuse opportunities do not exist
  - multiple disparate ANNs means limited reuse opportunities and thus SRAM is of limited use
  - no or limited batch processing

Solution

- 3DIC Architecture
  - reduces energy and area
  - increase connectivity and bandwidth
- 3D-DRAM
  - provide high bandwidth and large storage
  - operate directly out of DRAM
- Data Structures
  - data structures to ensure neural network data can be accessed efficiency
- Specialized processing layers
  - provide special functions to aid in acceleration of target neural networks



Solution

- Can a 3D-DRAM be used effectively?
- Can a useful system fit within the 3D stack?
- How can we control such a system?

Contribution

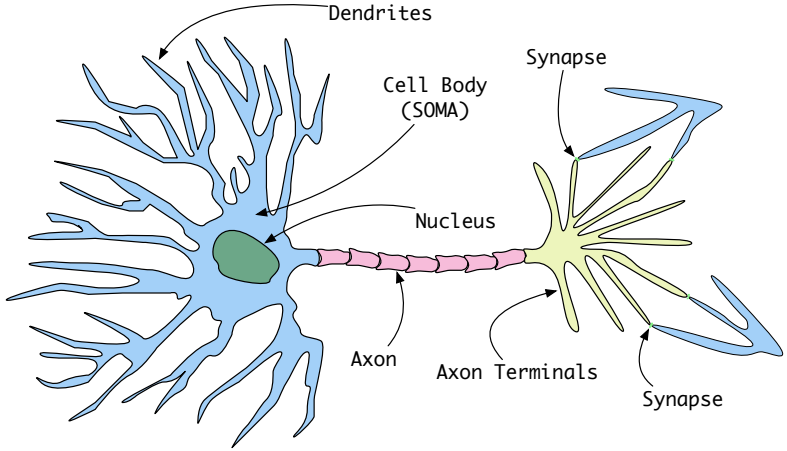
- An extensible architecture that can simultaneously process multiple disparate real-time DNNs
  - with low power and real estate demands
- Proposed a custom 3D-DRAM providing a ~32X bandwidth benefit compared to standard 3D-DRAM
  - wider bus and expose more of the page
- A DNN system solution that employs pure 3DIC technology
  - providing power and performance benefits of remaining within a 3DIC stack
- Custom instructions and data structures that facilitate operating directly out of 3D-DRAM
  - provide the required processing bandwidth by ensuring effective use of the 3D-DRAM
  - instruction format allow system functions to operate concurrently

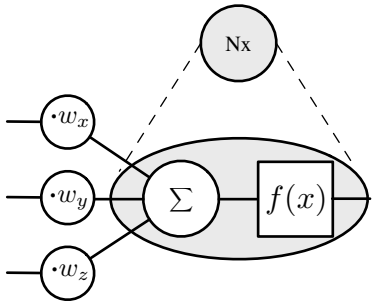
Outline

1. What are artificial Neural networks
2. Target application and ANN type's
3. State-of-the-art
4. System Architecture
   - Problem
   - Solution
5. System details
   - 3DIC DRAM and data structures
   - 3D Bus
   - Manager
   - Processing engine
6. Results
7. Summary

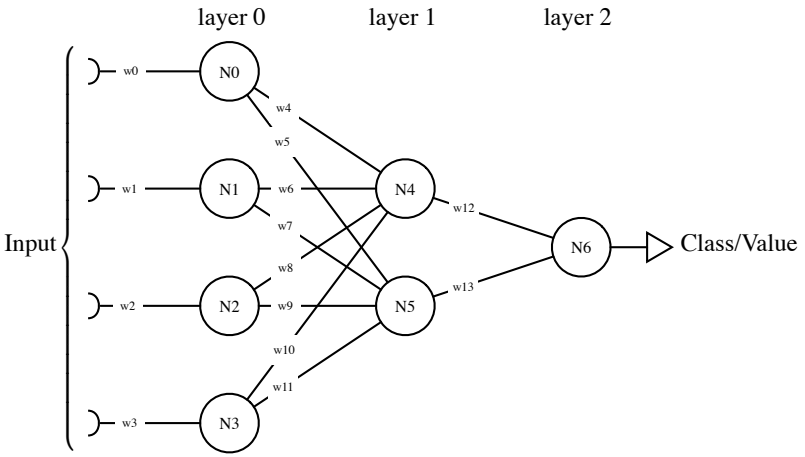
Artificial Neural Networks

- a network of processing elements inspired by the connectivity and processing observed in the brain





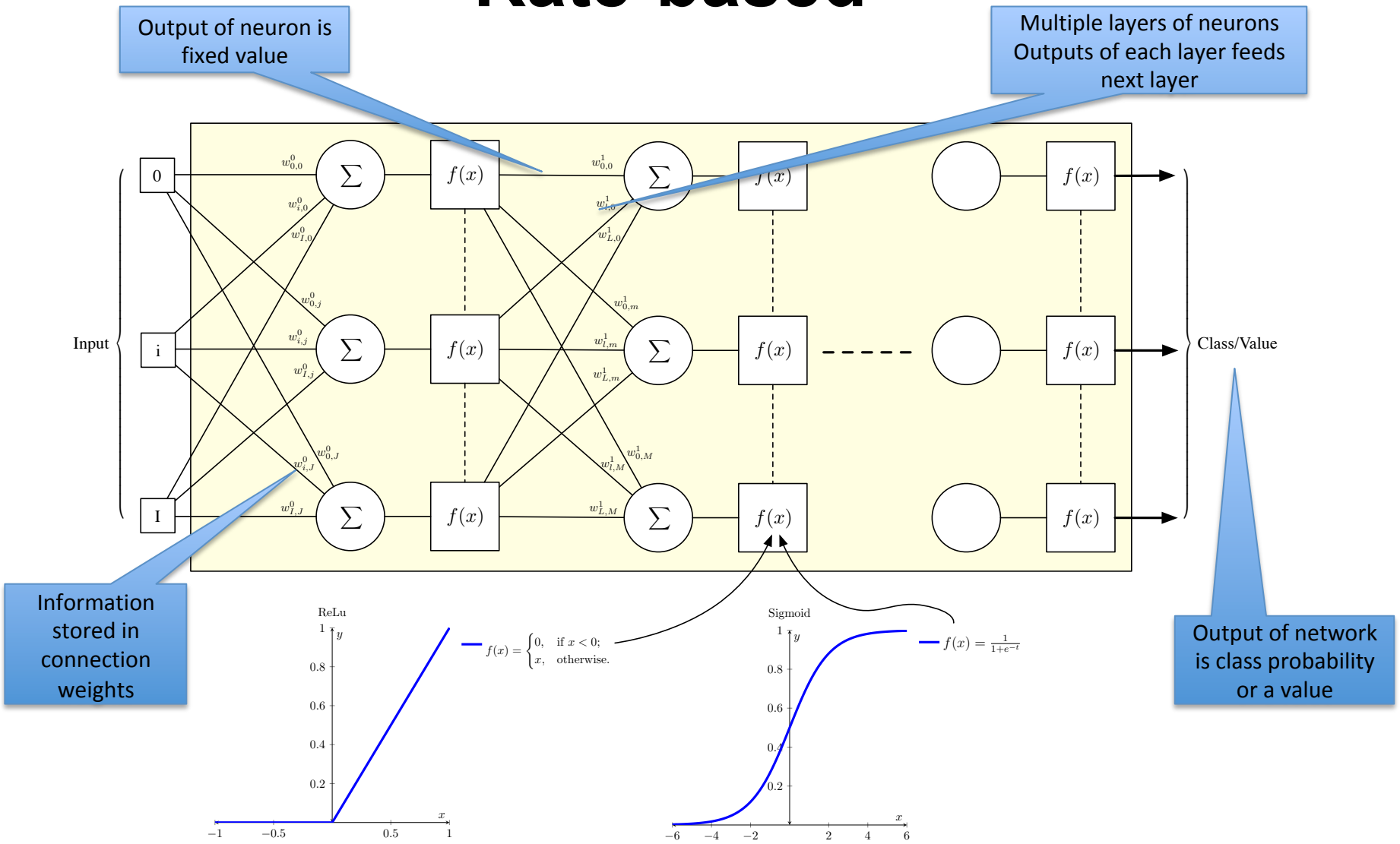
- the processing elements or neurons are connected together to form a network



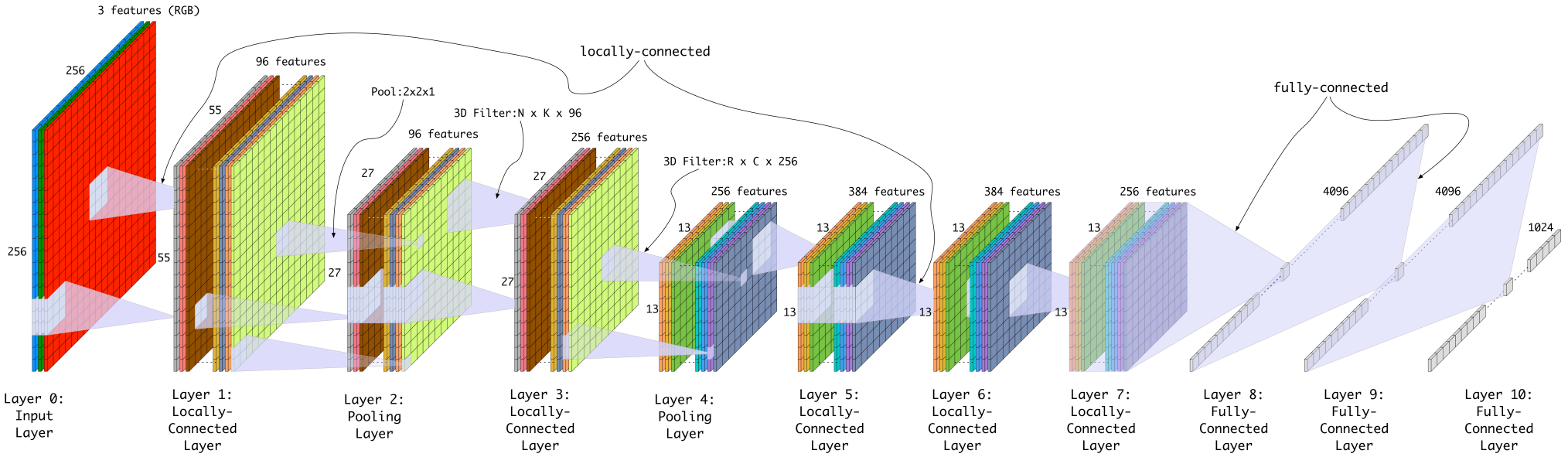
Artificial Neural Networks

- the processing elements fall into two categories
  - rate based neurons which try to capture the neuron behavior in the form of a number
    - relatively simple to model
    - information stored in the form of strength of connections
    - easier to train and have shown high levels of efficacy in many applications
  - spiking neurons which more closely emulate actual brain behavior
    - model neural activity in the form of differential equations
    - require numerical methods to solve although there have been attempts to employ analog circuits
    - information stored in the form of strength and delay of connections
    - complex and hard to train
- In this work we are focusing on rate based ANNs

Artificial Neural Networks Rate-based



Artificial Neural Networks
Rate-based

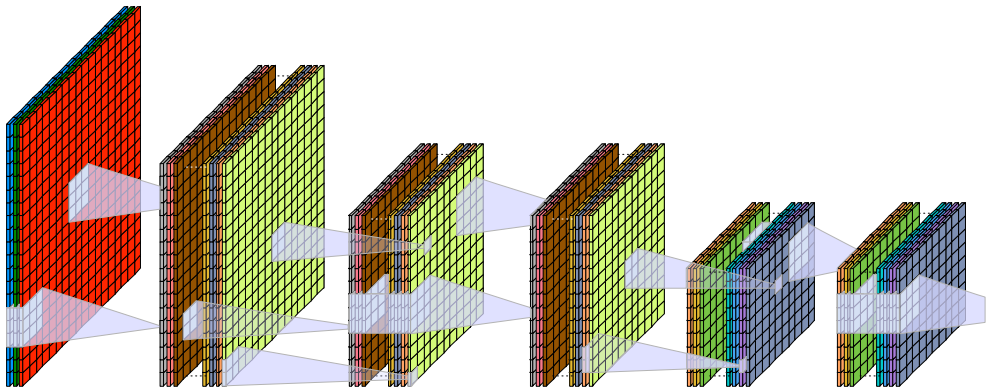


Summary : Target application and ANN type's

- Embedded systems
  - not cloud based
  - assumes multiple tasks are being performed
  - requires multiple disparate ANNs
- This work has focused on the DNNs :
  - Classifiers
  - function approximators
  - CNNs and DNNs
  - can be extended to support LSTMs

Deep Neural Network Classifiers

- Currently the most popular ANN
  - trained offline
  - Convolutional NN's have been very successful in classification
  - locally-connected NN's also used in face recognition
  - fully-connected includes LSTM
- This work focuses on inference in embedded applications



State-of-the-art

- Current implementations either use GPU's or low capacity ASIC implementations
- GPU's are typically general purpose devices and consume high power
  - GPU's do lend themselves to convolution when weights are shared
  - ASICs almost always provide performance iprovements
- ASIC implementations target larger networks by using multiple devices
  - usually assume local SRAM which limits network size
  - some/most implementations target specific NN's, usually CNNs

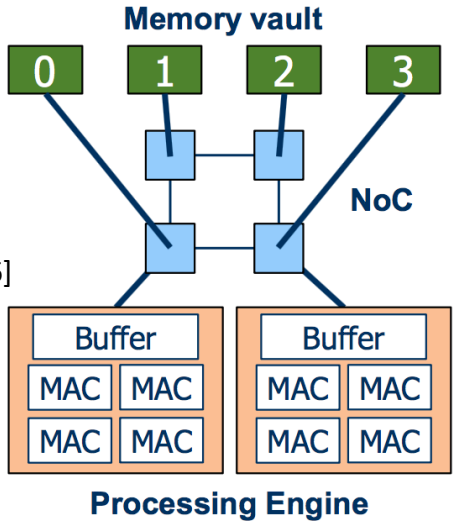
State of the Art

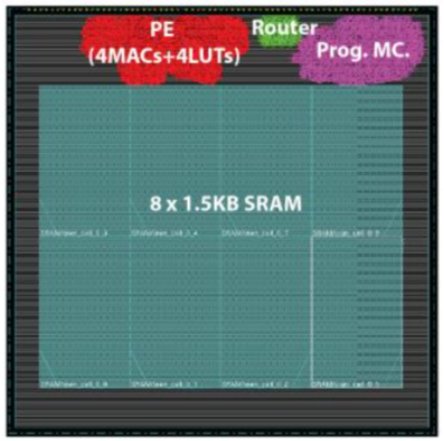
- ASICs
  - Most employ arrays of processors
  - Some target specific ANNs
  - Most cannot support useful sized NNs with single device
  - Many process data from local SRAM memory which often consumes >60% of silicon area
  - demonstrate performance improvements over GPUs

State of the Art ASIC's

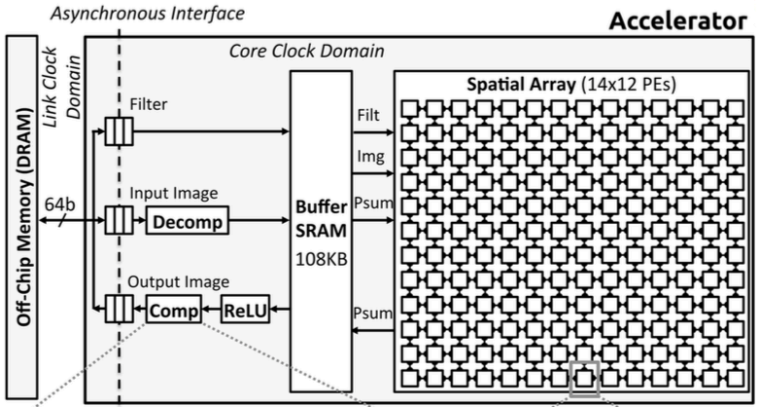
- ASIC's

NeuroCube[Kim16]





Eyeriss[Che16]



State of the Art ASIC's

- NeuroCube[1]
  - uses HMC 3D-DRAM but dependence on SRAM limits support to CNNs
- Eyeriss[2]
  - discusses DRAM accesses but their focus is on the convolution operation
  - supports CNN convolution operation only
- NnSP[3]
  - uses cache between DRAM and 8Mb SRAM and will be limited by DRAM bandwidth
  - does not discuss multi-devices
- Neuflow[4]
  - limits their support to CNNs and in the case of Eyeriss only supports the convolution
  - will not support locally-connected ANNs

[1] [Kim16], [2] [Che16], [3] [Esm05], [4] [Far11]

More recent State of the Art ASIC's

- NeuroStream
  - acknowledges DRAM is required for useful DNNs
  - uses HBM 3D-DRAM along with local SRAM
  - depends on SRAM for performance
- Google TPU
  - assumes large levels of batch processing
  - 8-bit processing
  - fully-connected NNs represent very high percentage
  - acknowledges performance degradation using fully-connected
- DaDiannao
  - uses embedded DRAM but still requires up to 64 for useful DNNs
  - high internal bandwidth but dissipates high power when scaling to useful-sized ANNs

[1] [Kim16], [2] [Che16], [3] [Esm05], [4] [Far11]

State of the Art
GPU's

- GPU's target the general market and employ infrastructures not necessary for NN implementation
  - we associate power of GPU solution ~100-200W
  - Will still be limited by memory bandwidth although they are starting to utilize 3D-DRAM
  - general purpose architecture makes it hard/impossible to make full use of theoretical performance

	CPU	V6	mGPU	IBM	GPU
Peak GOPs	10	160	182	1280	1350
Real GOPs	1.1	147	54	1164	294
Power W	30	10	30	5	220
GOPs/W	0.04	14.7	1.8	230	1.34

Table 5. Performance comparison. 1- CPU: Intel DuoCore, 2.7GHz, optimized C code, 2- V6: neuFlow on Xilinx Virtex 6 FPGA—on board power and GOPs measurements; 3- IBM: neuFlow on IBM 45nm process: simulated results, the design was fully placed and routed; 4- mGPU/GPU: two GPU implementations, a low power GT335m and a high-end GTX480.

[Far11]

State of the Art

- Current state-of-the art ASIC's and GPU's do not provide the capacity and efficiency to provide a solution requiring multiple useful disparate ANN's to be computed in real-time

Problem

- System solution requiring multiple NN's
  - ~8GB of memory
  - real-time processing : ~16mS (60 frames/sec)
  - ~26Tbps memory bandwidth
- Current 3D-DRAM memory technology bandwidth
  - HMC – SERDES~2Tbps
  - HBM – wide DDR ~2Tbps
  - Tezzaron DiRAM4 – wide DDR – 4Tbps
- Goldilocks principle
  - need balance of bandwidth and storage
  - currently bandwidth too low

Solution

- Architecture includes:
  - Custom organized 3D-DRAM specification
  - Data Structures specific to each ANN
  - Management Layer for configuration and control
  - Processing layer(s) targeted toward a family of ANNs
  - array of processing engines (PE) with streaming functions
    - multiply/accumulates ANe activation
    - multiply for softmax
  - additional special functions used in the neuron activation process
    - ReLu for ANe activation
    - divide and exponent for softmax function
    - compare for pooling

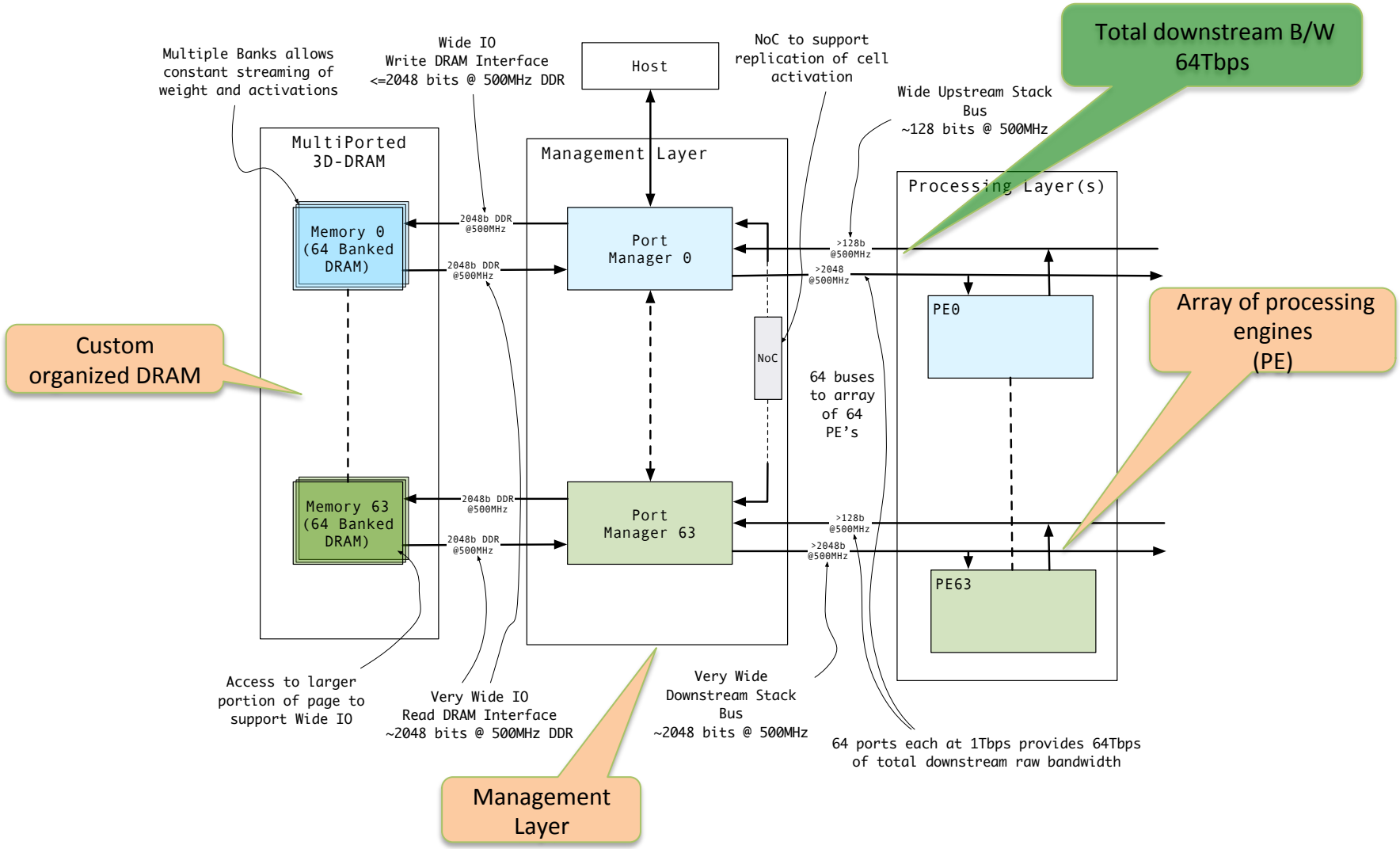
Solution

- 3D-DRAM is the Tezzaron DiRAM4
  - 64 disjoint 1Gb memory ports
  - System has 64 sub-systems each operating on one memory port
- Suggested Customizations for a 3D-DRAM
  - Standard DiRAM4 port is 32bits @ 1GHz
  - We suggest widen to 2048 bits and use high-density TSVs
  - Entire page in one access using burst-of-2
  - Raw bandwidth ~2Tbps per port

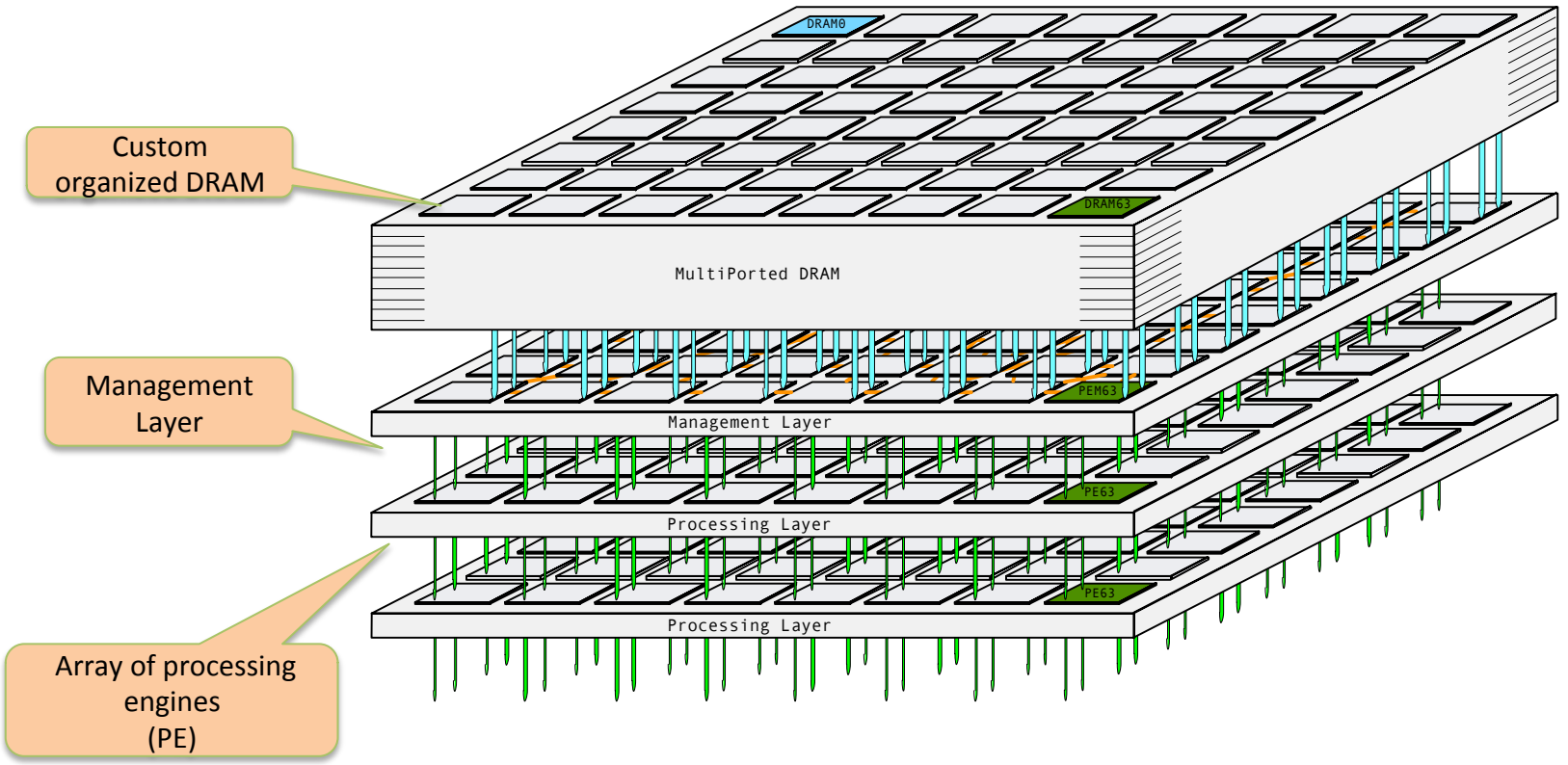
Solution

- System has 64 sub-systems each operating on one memory port
  - ANNs split over sub-systems
- Sub-system has a Manager and PE
  - Data-path between DRAM, Manager and PE
    - data for 32 execution lanes with two arguments per lane
    - able to perform a MAC operation between a weight and ANe state every cycle
    - process a group of 32 ANes each instruction
  - Result path from PE to Manager
    - carries result(s) of processing a group of ANes
    - return data contains up to 32 words
  - Instruction Decode in Manager
    - contains information on how to process a group of Anes
    - where are weights, ANe states and result stored

Solution Block Diagram

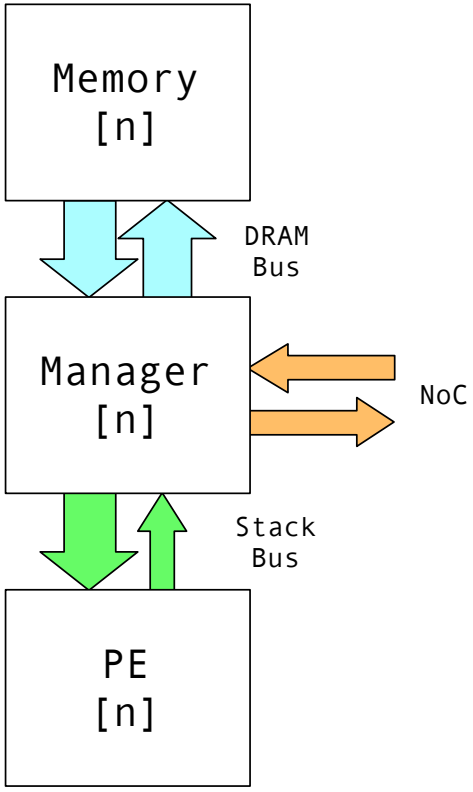


3D Physical Configuration

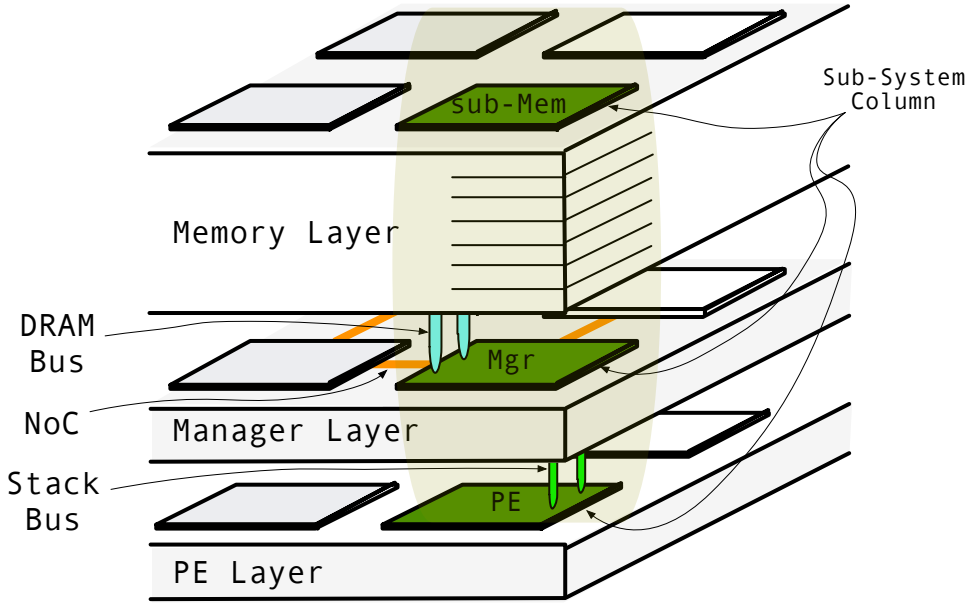


Sub-System Column (SSC)

- Consists of DRAM, manager and processing engine (PE)

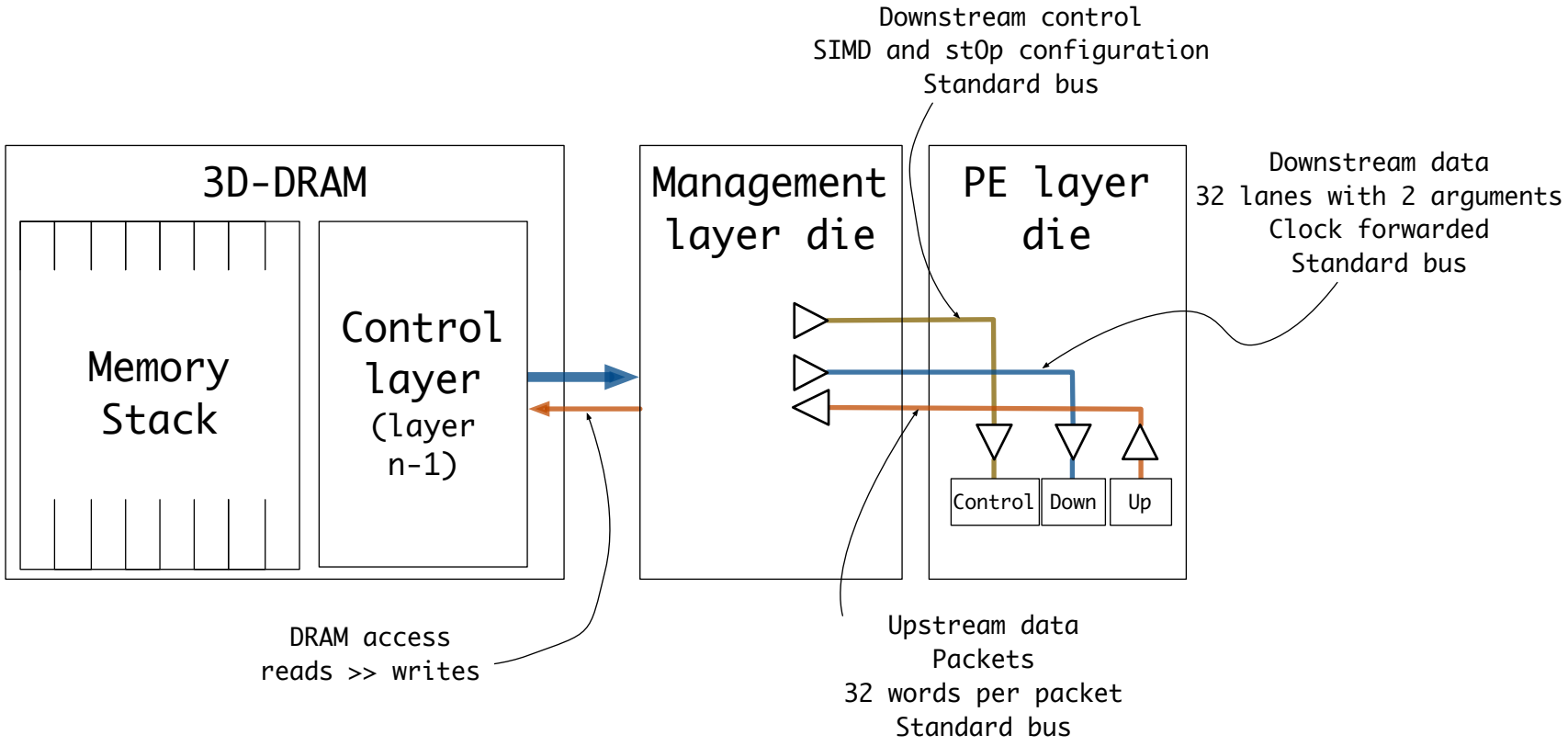


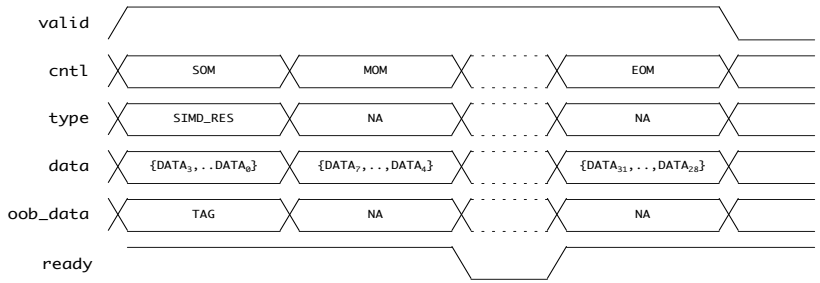
logical



physical

3D Stack Bus


Downstream control
SIMD and stOp configuration
Standard bus
3D-DRAM
Memory Stack
Control layer (layer n-1)
Management layer die
PE layer die
Downstream data
32 lanes with 2 arguments
Clock forwarded
Standard bus
Control
Down
Up
DRAM access
reads >> writes
Upstream data
Packets
32 words per packet
Standard bus


valid
cntl
SOM
MOM
EOM
type
SIMD_RES
NA
NA
data
{DATA3,..DATA0}
{DATA7,..,DATA4}
{DATA31,..,DATA28}
oob_data
TAG
NA
NA
ready

System

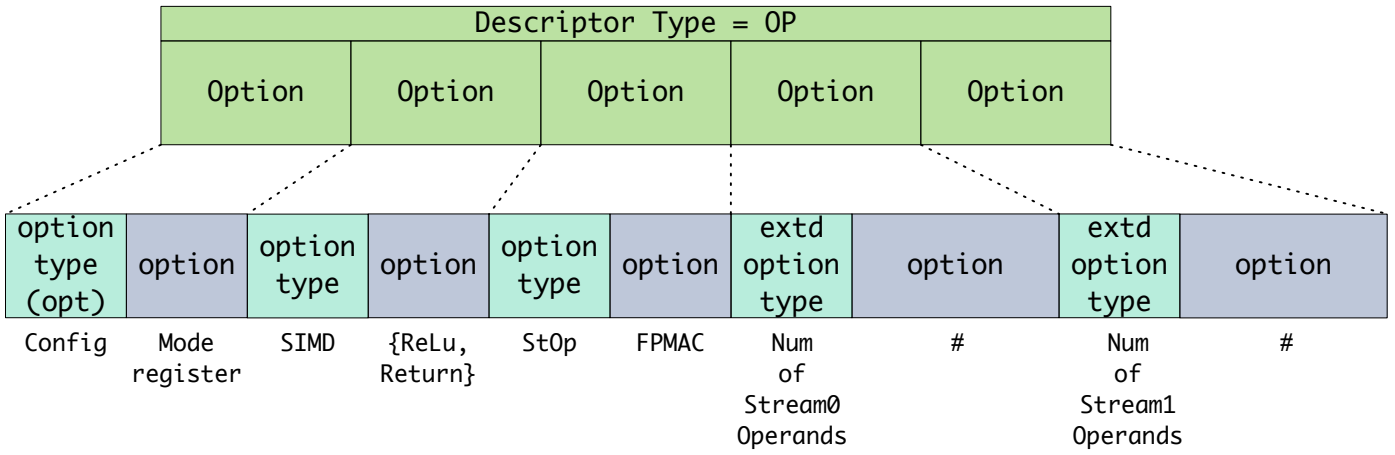
- Need to absorb DRAM latency
  - separate memory requests from read/write data
- Instruction spawns multiple commands to dependent tasks
  - concurrently pre-fetch data, prepare PE, prepare Return Data Processor
  - all operations have an associated tag
- Need to describe parameter and state storage
  - assume input stored in row-major fashion
- Network-on-Chip
  - needed for ANe state replication to all dependent SSCs

Instructions

- Instructions consist of descriptors

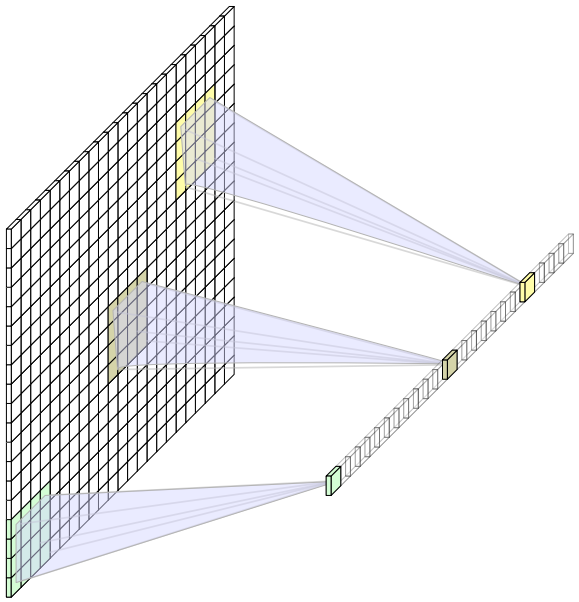
Operation Descriptor	arg0 Read Descriptor	arg1 Read Descriptor	Result Write Descriptor

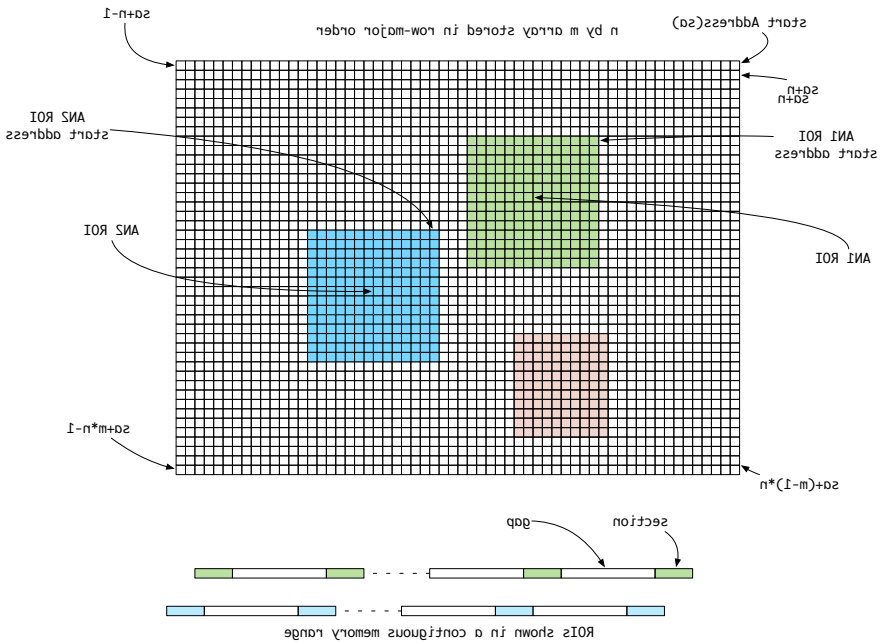
- Descriptors used to control modules which take part in the instruction
- Descriptors contain tuples which contain the details



Data Storage and access

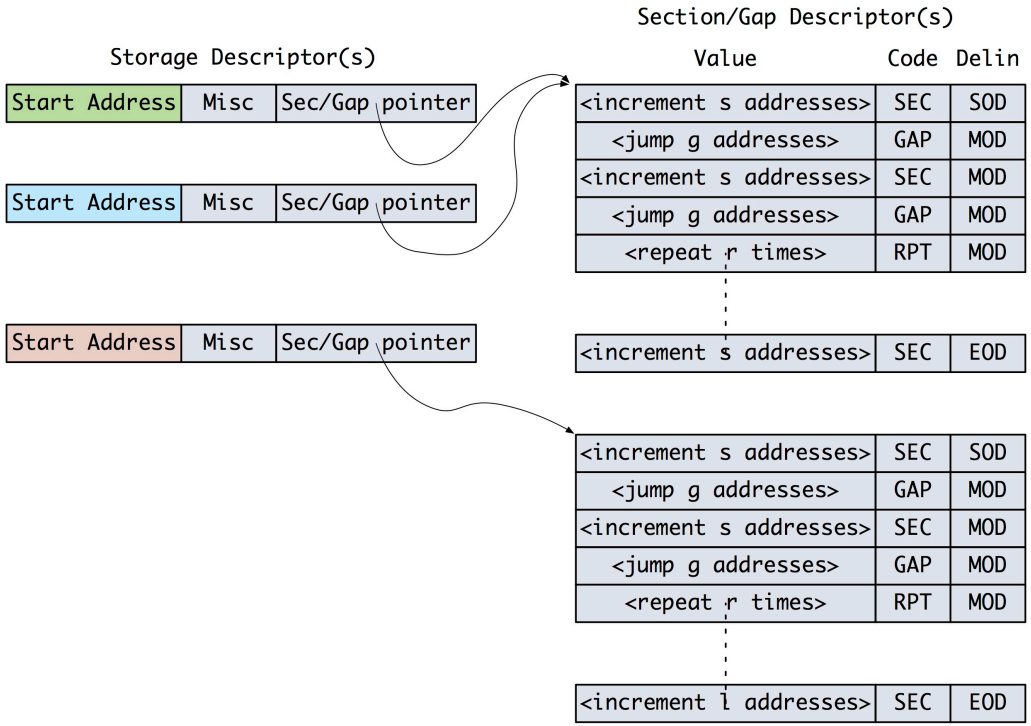
- Inputs and ANe states stored in row-major
- Weights for a group of ANes are interleaved





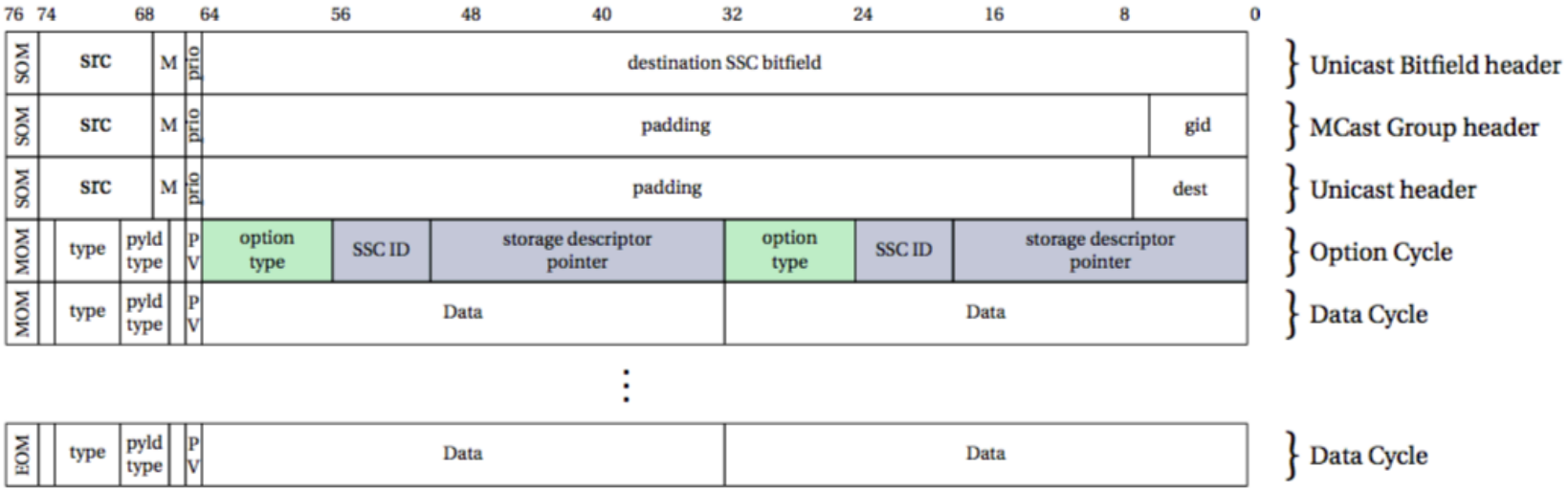
Storage descriptors

- Parameter and state storage
  - instruction contains pointer to storage descriptor
  - each SSC has a set of storage descriptors
  - instruction/NoC carry pointer, not address, length etc.



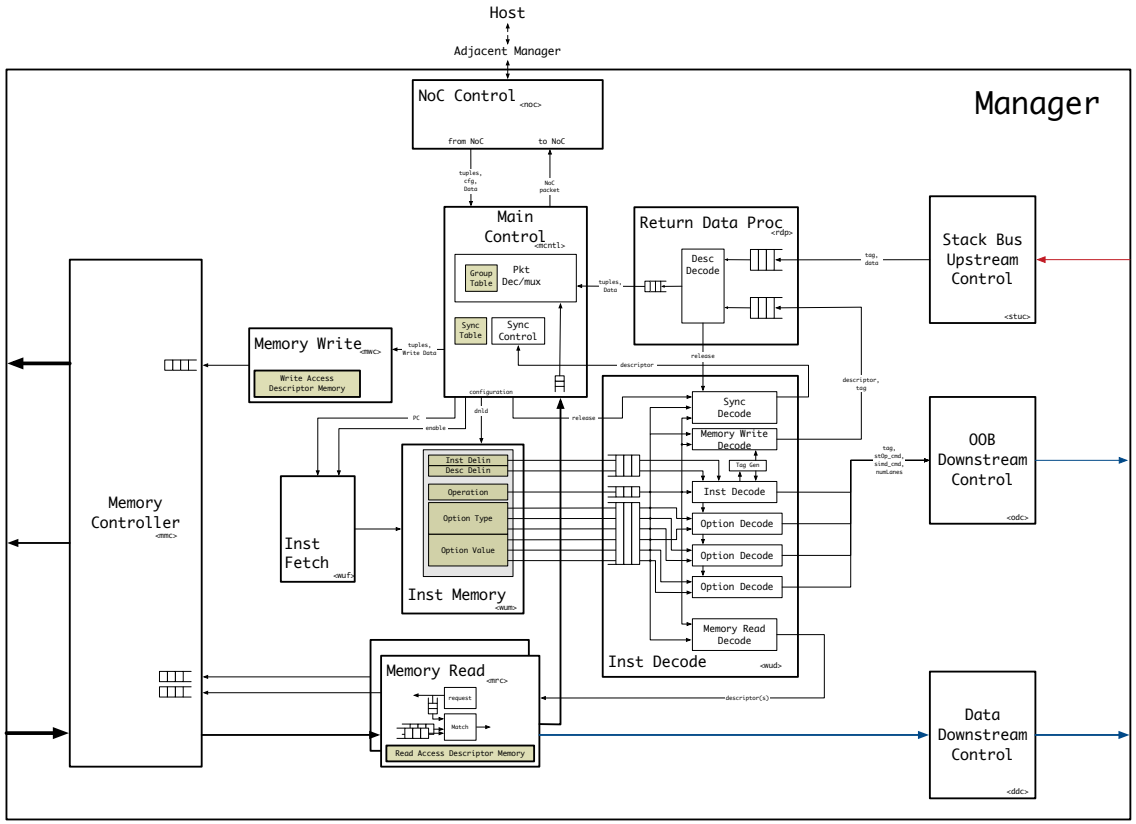
NoC

- Four ports
  - connections to adjacent SSC
  - fixed routing table
  - can be reconfigured based on performance
  - good area for additional research



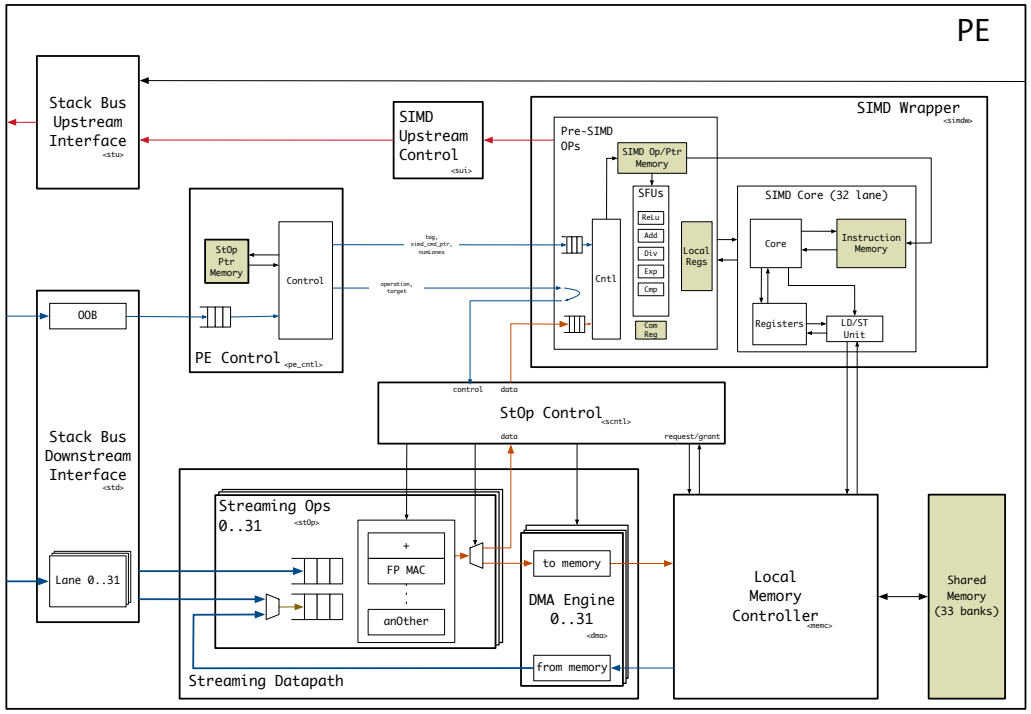
Manager Layer

- Decodes instructions
  - sub-descriptors are sent to dependent blocks
- Reads and writes to main memory
- Communicates to host and other SSCs



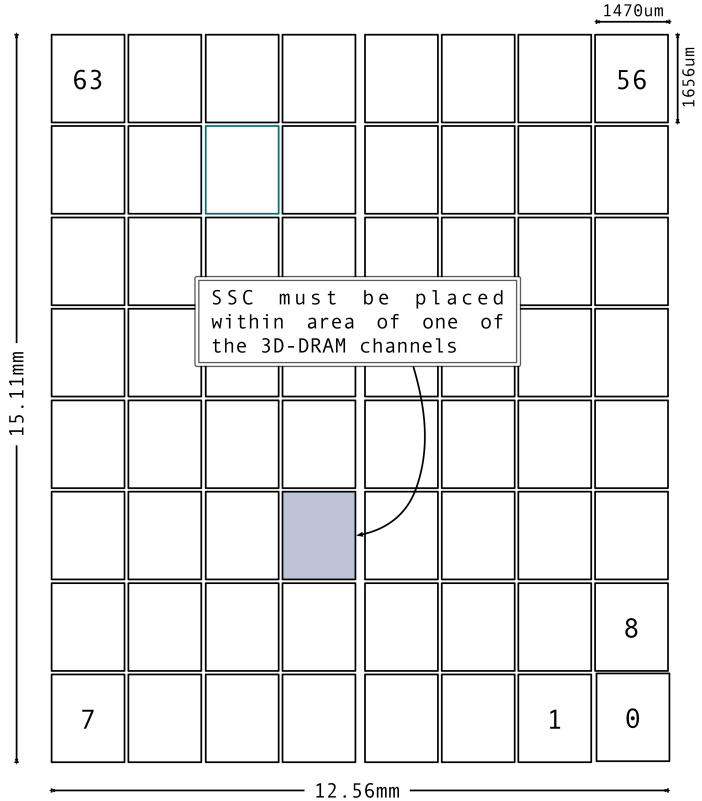
Processing (PE) Layer

- Streaming operations operate on data directly from the Stack bus
  - MAC, Multiply
- SIMD Wrapper performs post stOp tasks
  - add/divide/exponent, sends result back to manager



AREA

- Area based on DiRAM4
  - SSC needs to fit into footprint of DiRAM4 sub-memory
  - available area is 2.43 sq-mm



Performance

- Tested against multiple fanins

Test	System bandwidth (Tbps)
CONV2	22
CONV-294	23
CONV-300	25
CONV-500	26
CONV-1000	30
FC-350	26
FC-500	27
FC-1000	30
FC-7	31

- Expected baseline

Layer	fanin	Equivalent test	Percentage of Operations	Expected bandwidth
1	363	CONV-300	44%	10.9
2	4	previous layer		
3	2400	CONV-1000	28%	8.5
4	4	previous layer		
5	2304	CONV-1000	10%	3
6	3456	CONV-1000	10%	3
7	3456	CONV-1000	7%	2
8	43264	FC-7	1%	0.2
9	4096	FC-7	1%	0.2
10	4096	FC-7	<1%	0.05
			Total	>27.7

Scaling assumptions

- Scaling based on synthesizing representative block at 28nm and 65nm

- Area

Area Scaling 65nm -> 28nm	
Memory	2.79
Logic	2.68

- Power

	Synthesis numbers					
	65nm			28nm		
	Internal	Net switch	Leakage	Internal	Net switch	Leakage
Logic	66.9	1.53	2.02	13.2	1.26	4900
memory	2.36			0.0438		

	Scaling		
	65nm to 28nm		
	Internal	Net switch	Leakage
Logic	5.07	1.21	4.12E-04
memory	5.07		

AREA

- Manager/PE Block sizes

	Manager		
	Distribution	65nm (um)	28nm(um)
Memory Controller	20.6%	994000	363478
NoC	6.9%	335000	122500
Read Control	47.1%	2270176	830141
Write Control	6.7%	321000	117381
Instruction Proc	1.7%	83570	30559
Read data proc	1.6%	78000	28522
System Controller	1.6%	77000	28157
TSV	6.9%	333750	333750
Misc	6.8%	330000	120672
	100.0%	4822496	1975160

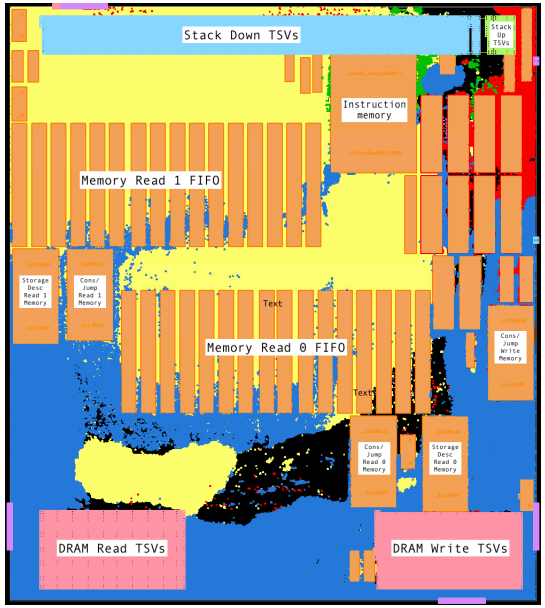
	Manager	
	65nm	28nm
Area used	4822496	1975160
Area available	6661496	2434320
Utilization	72%	81%
Utilization w/o TSV	71%	78%

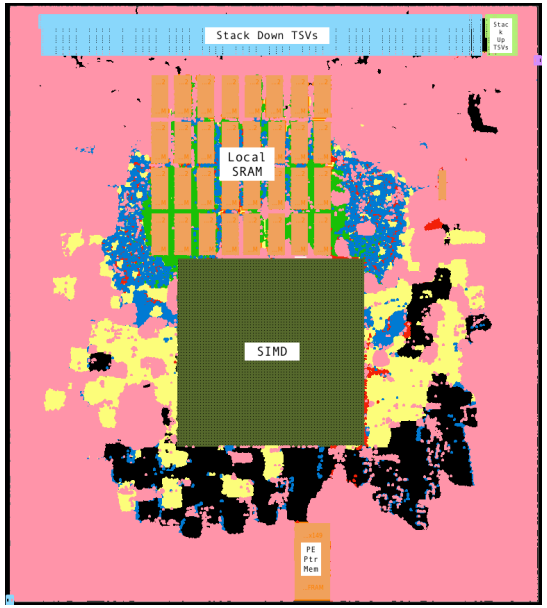
	Processing Engine		
	Distribution	65nm	28nm
Operation Decode	3.1%	105000	38370
Return Data Control	1.5%	49000	17906
SIMD Wrapper	15.1%	507000	185274
SIMD	17.9%	600000	219259
Streaming Ops	40.0%	1344000	491140
StOp Control	1.9%	64000	23388
LM	16.4%	550000	200987
TSV	3.7%	124750	124750
Misc	0.4%	15000	5481
Total	100.0%	3358750	1306555

	Processing Engine	
	65nm	28nm
Area used	3358750	1306555
Area available	6661496	2434320
Utilization	50%	54%
Utilization w/o TSV	49%	51%

Area placement study

- Placed and routed without DRC or LVS
- Routing congestion showed minimal hotspots
- Parasitics used in Primetime for power analysis





Power Summary

- Power estimates based on CONV-294 testcase simulation
  - back to back operations
  - accumulate DRAM access and use DiRAM4 datasheet [xx]
  - TSV energy from [XX]
  - parasitics from layout
  - designed using 65nm library
  - 28nm is target technology
  - power and area scaling based on synthesizing representative blocks

Parameters	
Frequency	500MHz
Test	CONV-294

Block	Power
Manager	42.55
PE	26.50
DRAM	4.51
DRAM TSVs	1.14
Stack Bus TSVs	0.74
Total	**75.44**

Research

- Specify a custom organization of a 3DIC DRAM
  - provide a specification showing memory organization, bus widths
- Create Structures/Instructions to exploit DRAM capacity
- Propose 3D architecture to take advantage of DRAM capacity and bandwidth
  - RTL design and simulate memory management layer
  - RTL design and simulate processing layer

Comparison

- Best comparison is to NeuroStream and DaDianNao
  - Goldilocks

	TPU	NeuroStream	DaDianao	this work
Bandwidth to Capacity Ratio	5	32	8619	406
	Cold	Cold	Hot	Just right

  - Scaled

	TPU	NeuroStream	DaDianao
Power at capacity	40	88	3549
Ratio	53%	117%	4732%
Power at bandwidth	3611	1117	167
Ratio	4815%	1490%	223%
Area at capacity	300	792	14889
Ratio	86%	226%	4254%

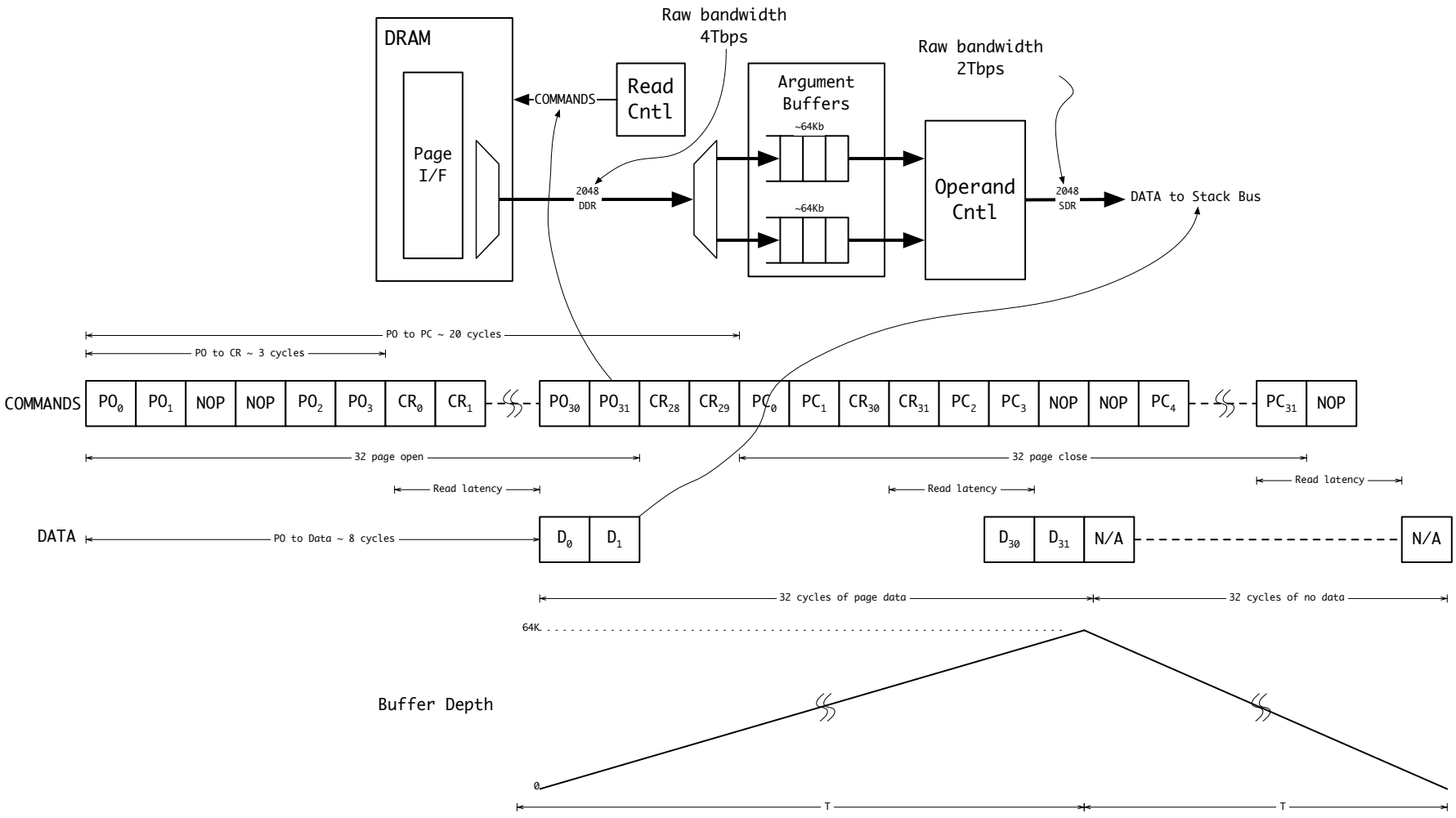
Summary

- Real world applications will require multiple artificial neural networks
  - current solutions consume significant power and real-estate
- Proposed a 3DIC solution that includes:
  - Custom organized 3D-DRAM specification
  - Instructions and data structures
  - Purpose designed functions to accelerate a Artificial Neural Networks
  - Architecture to take advantage of 3D technology
- Overall performance and power improvement

References

Backup

Example DRAM Access Sequence

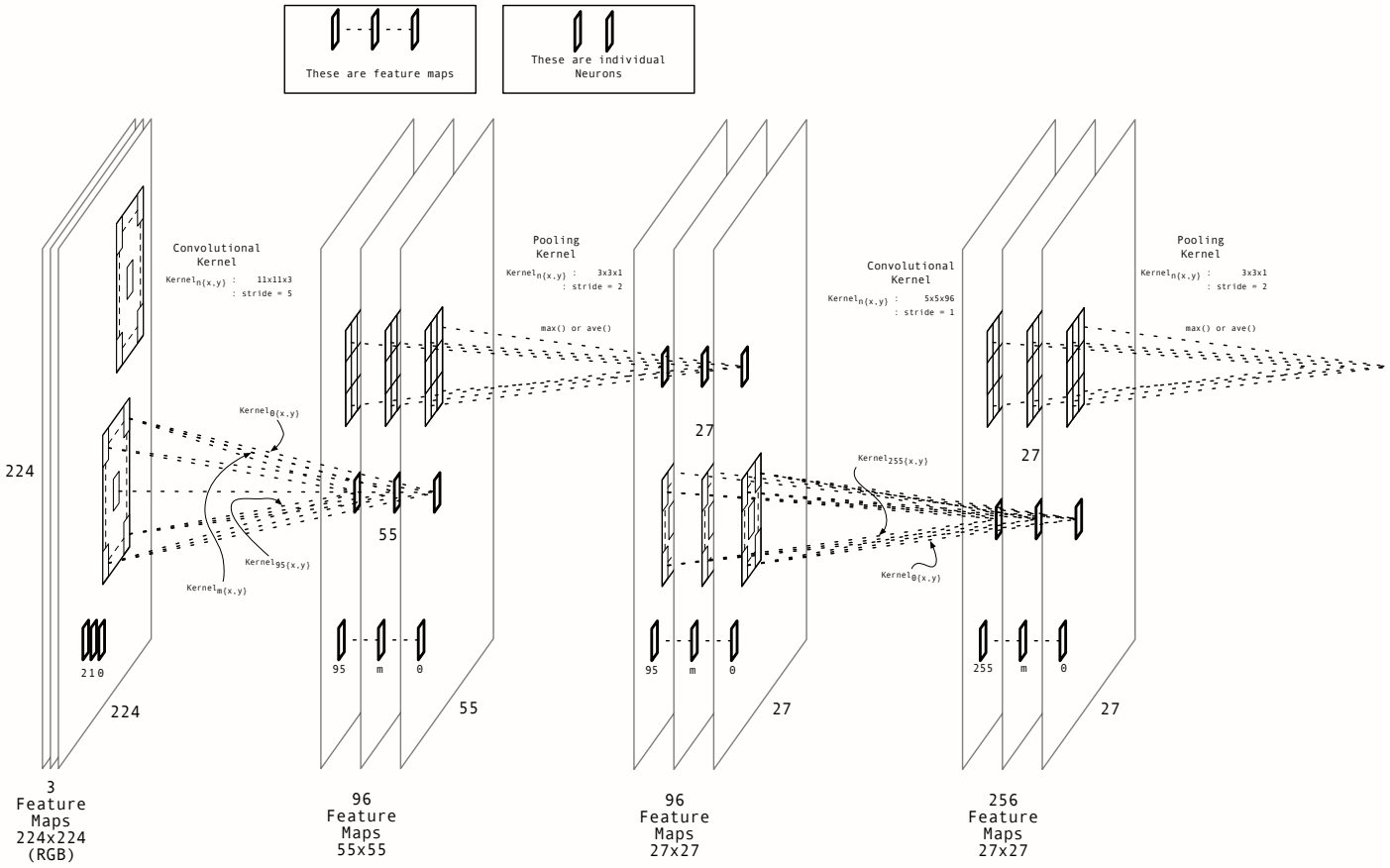


Stack Bus

- Downstrean and upstream buses
  - OOB bus carries control
- Buses carry control packets and data packets
  - packets will be multiple clock cycles
- Control packets used to configure PE
  - special function and SIMD operations
  - result information
- Data Packets will carry arguments for special functions
  - data to will be multiple cycles of two 32-bit words

Example Image Recognition CNN

- Example taken from Krizhevsky, Sutsever, Hinton 2012 [Kri12]
- Seven layers but only first 3 layers shown
- ~60M parameters(~2Gb), ~2GFLOP



TSV Power

from [Liu12]

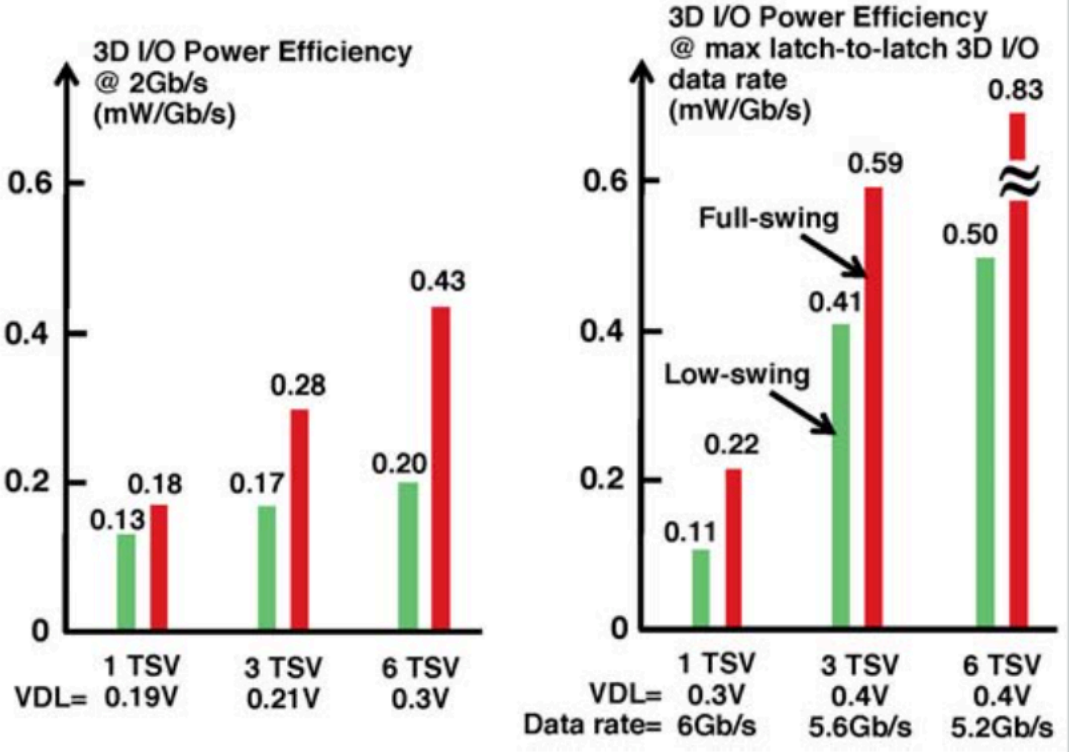
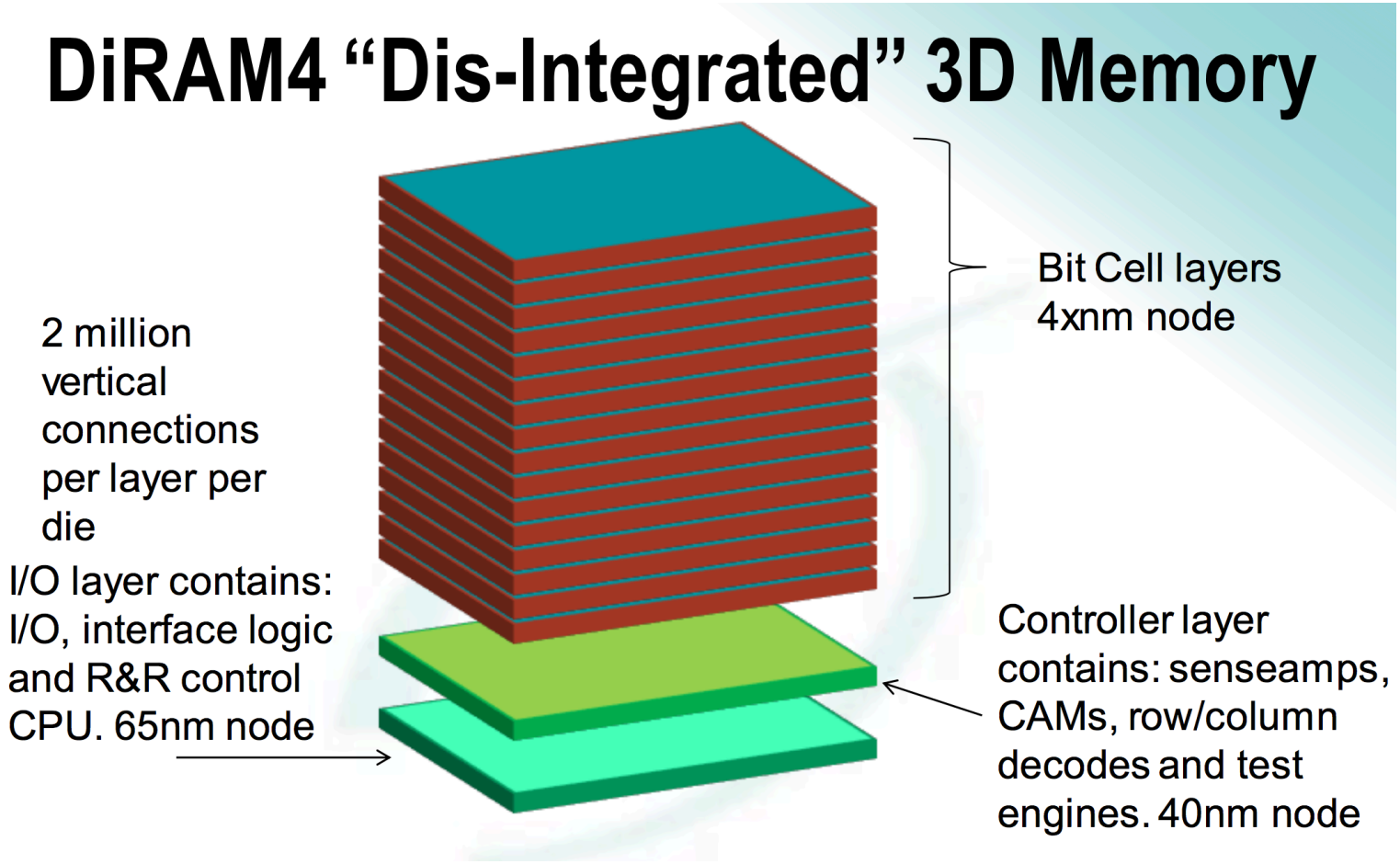


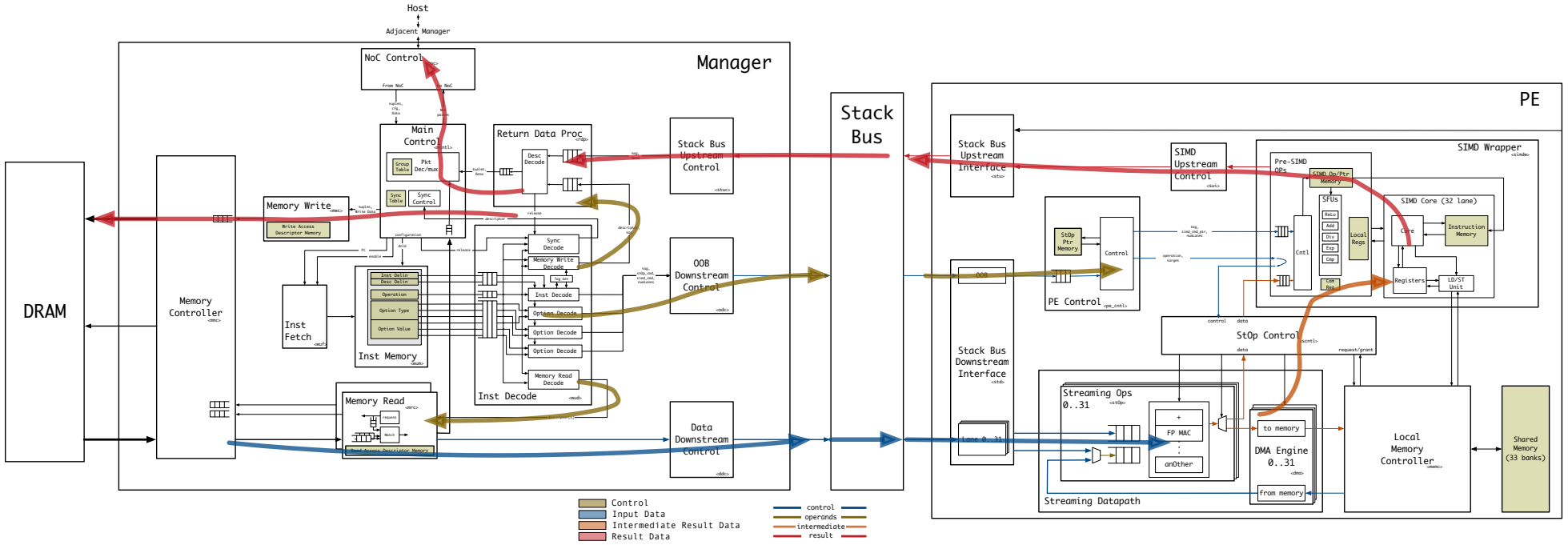
Figure 7.7.5: Measured power efficiency comparison of low-swing and full-swing latch-to-latch 3D I/O test sites.

DiRAM4 Stack

from [Pat14]



Detailed Block Diagram



Scaling from [Sch14]

THE SCALING FACTORS UTILIZED IN THIS WORK.

Technology	130	90	65	45	32	21	15	10.6	7.5
ITRS [7]									
V_{dd}	1.34	1.13	1.13	1	0.93	0.84	0.75	0.68	0.62
Frequency	0.33	0.41	0.61	1	1.24	2.46	3.15	3.93	4.75
Capacitance	4.75	3.17	1.73	1	0.7	0.31	0.2	0.12	0.07
Energy	8.53	4.05	2.21	1	0.61	0.22	0.11	0.06	0.03
Dynamic Power	2.81	1.66	1.35	1	0.75	0.54	0.28	0.17	0.11
Leakage Power	5.66	1.21	3.41	1	1.46	1.07	0.68	0.45	0.29
Area	8	4	2	1	0.66	02	0.1	0.06	0.03
PTM [13]									
V_{dd}	1.3	1.2	1.1	1	1	0.9	0.8	0.75	0.7
Frequency	0.57	0.72	0.85	1	1.75	12.93	47.34	67.64	36.12
Capacitance	5.45	3.12	1.47	1	0.65	0.6	0.3	0.65	1.42
Energy	9.21	4.49	1.78	1	0.65	0.49	0.19	0.37	0.7
Dynamic Power	5.25	3.23	1.51	1	1.14	6.28	9.09	24.73	25.13
Leakage Power	3.2	1.18	3.75	1	0.84	0.13	0.08	0.05	0.03
Area	8	4	2	1	0.5	0.26	0.11	0.06	0.03
Intel [12]									
V_{dd}	1.49	1.26	1.11	1	0.93	0.88	0.86	0.84	0.84
Frequency	0.58	0.72	0.87	1	1.1	1.19	1.25	1.3	1.34
Capacitance	2.37	1.78	1.33	1	0.75	0.56	0.42	0.32	0.24
Energy	5.26	2.83	1.64	1	0.65	0.43	0.31	0.23	0.17
Total Power	3.05	2.03	1.43	1	0.71	0.52	0.39	0.29	0.23
Area	2.37	1.78	1.33	1	0.75	0.56	0.42	0.32	0.24

Estimated DRAM Internal Power

- Considering the DRAM Read path sequence shown previously (energy numbers from [Tez])

DiRAM4 Energy	**DiRAM Read Path Access Sequence**
- Page Open = 100pJ	- Number of Page Opens = 32
- Page Close = 320pJ	- Number of Page Closes = 32
- Page Read/Write = 64pJ	- Number of Page Reads = 32
- NOP = 20pJ	- Number of NOPs = 32
- Page Refresh = 320pJ	- Number of Cycles = 128

- Internal DRAM Power

$$\begin{aligned}\text{Average energy per cycle per port}(pJ) &= \frac{\#\text{PO's} \cdot E_{po} + \#\text{PC's} \cdot E_{pc} + \#\text{Read's} \cdot E_{cr} + \#\text{NOP's} \cdot E_{no}}{\text{Total number of cycles}} \\ &= \frac{32 \cdot 100 + 32 \cdot 320 + 32 \cdot 64 + 32 \cdot 20}{128} \\ &= 126\,\text{pJ}\end{aligned}$$

$$\begin{aligned}\text{Maximum Power} &= \frac{energy}{cycle} \cdot \frac{cycles}{sec} \cdot ports \\ &= 126 \times 10^{-12} \cdot 1 \times 10^{9} \cdot 64 \\ &= 8.06\,\text{W}\end{aligned}$$

- Our multi-CNN will likely use 54Tbps out of 131Tbps

$$\text{Actual Power} \approx \frac{54\text{Tbps}}{131\text{Tbps}} \cdot 8.06 = 3.32\,\text{W}$$

Stack Bus TSV Power

- Assumptions:
  - Assume Stack bus employs an invert bit to minimize transitions between cycles
  - TSV energy based on [Liu12]

$$\begin{aligned}\text{Maximum Power per Stack} &= \frac{J}{bit/cycle} \cdot \frac{ports}{stack} \cdot \frac{cycles}{sec} \cdot \left(ActivityFactor \cdot \frac{Databits}{port} + \frac{Clockbits}{port} \right) \\ &= 0.2 \times 10^{-12} \cdot 64 \cdot 1 \times 10^{9} \cdot (0.5 \cdot 2128 + 32) \\ &= 13.6 + 0.41 = 14.01\,\text{W per 4 die stack}\end{aligned}$$

$$\begin{aligned}\text{Expected Power per Stack} &= \left(\frac{\text{CNN Bandwidth}}{\text{Raw Bandwidth}} \cdot \text{Maximum Data Power} + \text{Clock Power} \right) \\ &\approx \frac{54\text{Tbps}}{131\text{Tbps}} \cdot 13.6 + 0.41 = 5.61 + 0.41 \\ &\approx 6.02\,\text{W}\end{aligned}$$

DRAM Page Bus TSV Power

- Assumptions:
  - Assume bus employs an invert bit to minimize transitions between cycles
  - Single TSV energy based on [Liu12]

$$\begin{aligned}\text{Maximum Page TSV Power} &= \frac{J}{bit/cycle} \cdot \frac{ports}{dram} \cdot \frac{cycles}{sec} \cdot \left(ActivityFactor \cdot \frac{Databits}{port} + \frac{Clockbits}{port} \right) \\ &\approx 0.13 \times 10^{-12} \cdot 64 \cdot 1 \times 10^{9} \cdot (0.5 \cdot 4128) \\ &= 17.2\,\text{W}\end{aligned}$$

$$\begin{aligned}\text{Expected TSV Power per Page Interface} &= \frac{\text{CNN Bandwidth}}{\text{Raw Bandwidth}} \cdot \text{Maximum Page TSV Power} \\ &\approx \frac{54\text{Tbps}}{131\text{Tbps}} \cdot 17.2 \\ &\approx 7.1\,\text{W}\end{aligned}$$

FMA Power/Area Estimate

- Consider an FMA running at the Stack bus speed of ~1GHz, from [GH11] table 1

Fused Multiply-Accumulate Power and Area [4]

Pipeline	Freq (GHz)	Area (um^2)	Static Power (mW)	Dynamic Power (mW)	$W/GFlop$	$mm^2/GFlop$	Power Density (W/mm^2)
6	2.08	16077	1.2	30.9	0.0077	0.0039	2.00
6	2.08	16077	1.2	30.9	0.0077	0.0039	2.00
5	1.32	14241	0.55	12.85	0.0051	0.0054	0.94
4	0.98	12670	0.58	8.09	0.0044	0.0065	0.68
3	0.5	12117	0.16	3.16	0.0033	0.0121	0.27
3	0.2	10619	0.0358	0.952	0.0025	0.0265	0.09

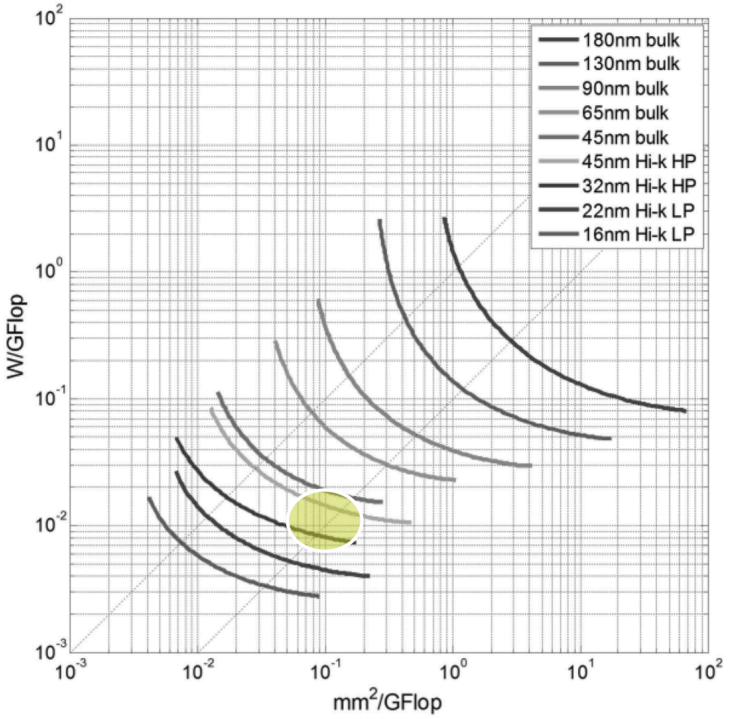
$$\text{FMA Area (45nm)} = area/fma \cdot \#PE \cdot fma/PE$$
$$= \frac{12670}{10^6} \cdot 64 \cdot 32 = 25.95\,\text{mm}^2$$

$$\text{Max FMA Power (45nm)} = \text{Number of FMA} \cdot (\text{leakage power} + \text{dynamic Power})$$
$$= 64 \cdot 32 \cdot \left(0.58 + \frac{8.09}{0.98} \cdot 1.0\right) = 1.187 + 16.91$$
$$= 18.1\,\text{W}$$

$$\text{Actual FMA Power (45nm)} \approx \frac{54\text{Tbps}}{131\text{Tbps}} \cdot 16.91 + 1.187 = 6.97 + 1.187 = 8.16\,\text{W}$$

FMA Power/Area Estimate

- From [GH11] figure 10 scaling from 45nm to 32nm whilst maintaining the mm^2/Gflop suggests a scaling factor of 0.53



$$\text{Actual FMA Power (32nm)} \approx \frac{0.8}{1.4} \cdot 8.16 = 4.66\,\text{W}$$

$$\text{FMA Area (32nm)} = 25.95\,\text{mm}^2$$

TSV Area

- Power Delivery
  - ~2000TSV's ~0.1mm^2
- Stack Bus
  - 2737 signals
  - Assume 2:1 for GND/VCC ~ 5500 TSV's
  - 5um pitch ~ 0.14mm^2/PE
  - 8.8mm^2 for 64 PE's
- DRAM Bus
  - 4255 signals ~ 8500 TSV's
  - 13.6mm^2 for 64 Ports

GPU Performance Backup

- from [Far11]

	CPU	V6	mGPU	IBM	GPU
Peak GOPs	10	160	182	1280	1350
Real GOPs	1.1	147	54	1164	294
Power W	30	10	30	5	220
GOPs/W	0.04	14.7	1.8	230	1.34

Table 5. Performance comparison. 1- CPU: Intel DuoCore, 2.7GHz, optimized C code, 2- V6: neuFlow on Xilinx Virtex 6 FPGA—on board power and GOPs measurements; 3- IBM: neuFlow on IBM 45nm process: simulated results, the design was fully placed and routed; 4- mGPU/GPU: two GPU implementations, a low power GT335m and a high-end GTX480.