Alireza Alivand

Department of Electrical and Computer Engineering,
Urmia University

B.Sc. Urmia University

BSc Project

Advisor: Mortaza Mojarad

Mentors: Amir Fathi & Morteza Mousazadeh

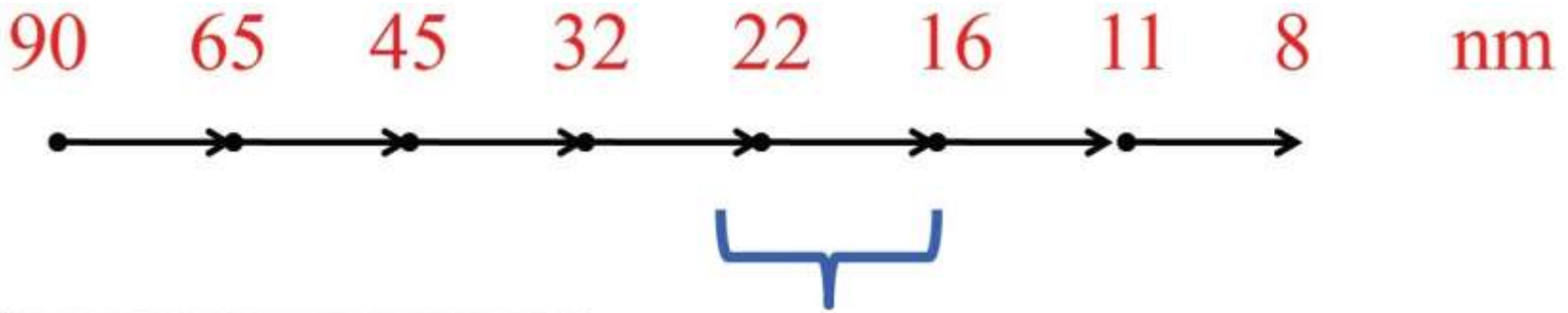
Spring 2021

Contents:

Acknowledgments

To my advisor, Mortaza Mojarad and my mentors Amir Fathi and Morteza Mousazadeh thank you for the opportunity to with you and learned from you, and for the advice, support and research insights.

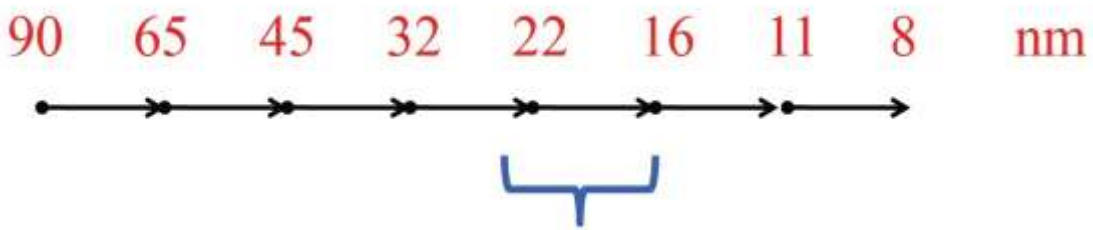
Scaling: Moore's Law



$$S = \frac{22}{16} = \sim 1.4x$$

Scaling: Moore's Law

- Scaling in Intel microprocessors
  - Mostly according to the Moor's law



$$S = \frac{22}{16} = \sim 1.4x$$

90 nm		Dothan
65 nm		Yonah
	Core	Merom
		Penryn
45 nm	Nehalem	Nehalem
32 nm		Westmere
	Sandy Bridge	Sandy Bridge
22 nm		Ivy Bridge
	Haswell	Haswell
14 nm		Broadwell
	Skylake	Skylake
		Kaby Lake
		Coffee Lake
10 nm		*Cannon Lake*
	Ice Lake	*Ice Lake*
7 nm or 10 nm++	Ice Lake or new architecture	*Tiger Lake*
		Sapphire Rapids

Scaling

Transistors scale as S^2

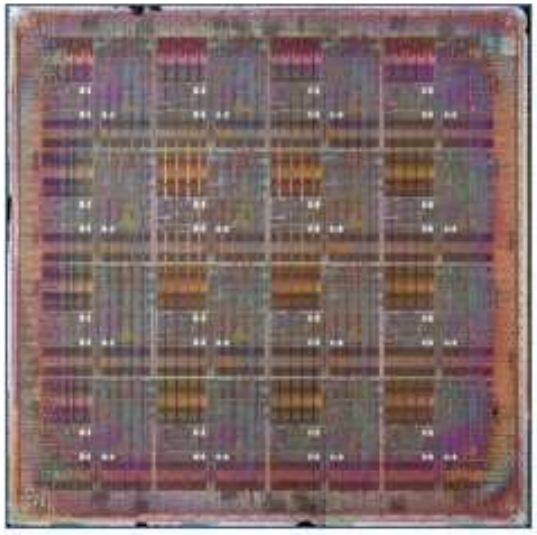
180 nm

16 cores

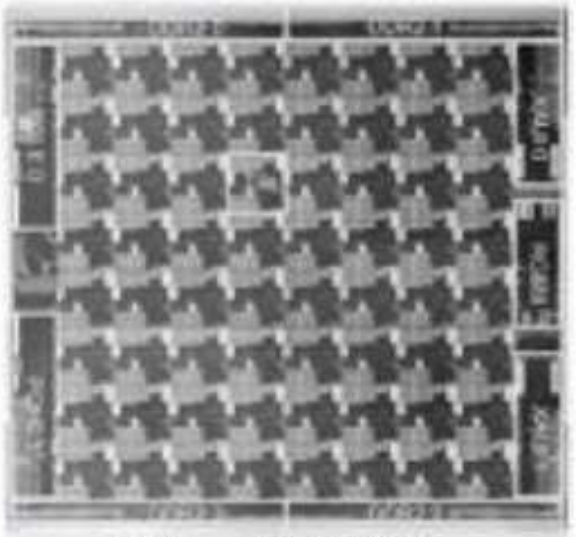
S = 2x
Transistors = 4x

90 nm

64 cores



MIT Raw



Tilera TILE64

Scaling- performance

- *S* is the scaling factor of transistors
  - S = 32nm/22nm = 1.4x between 32 and 22 nm process generations
- For each new technology, transistor count will scale by S^2, and transistor switching frequency will scale by S

→ performance scales by S^3 (2.8x from 32 nm to 22nm)

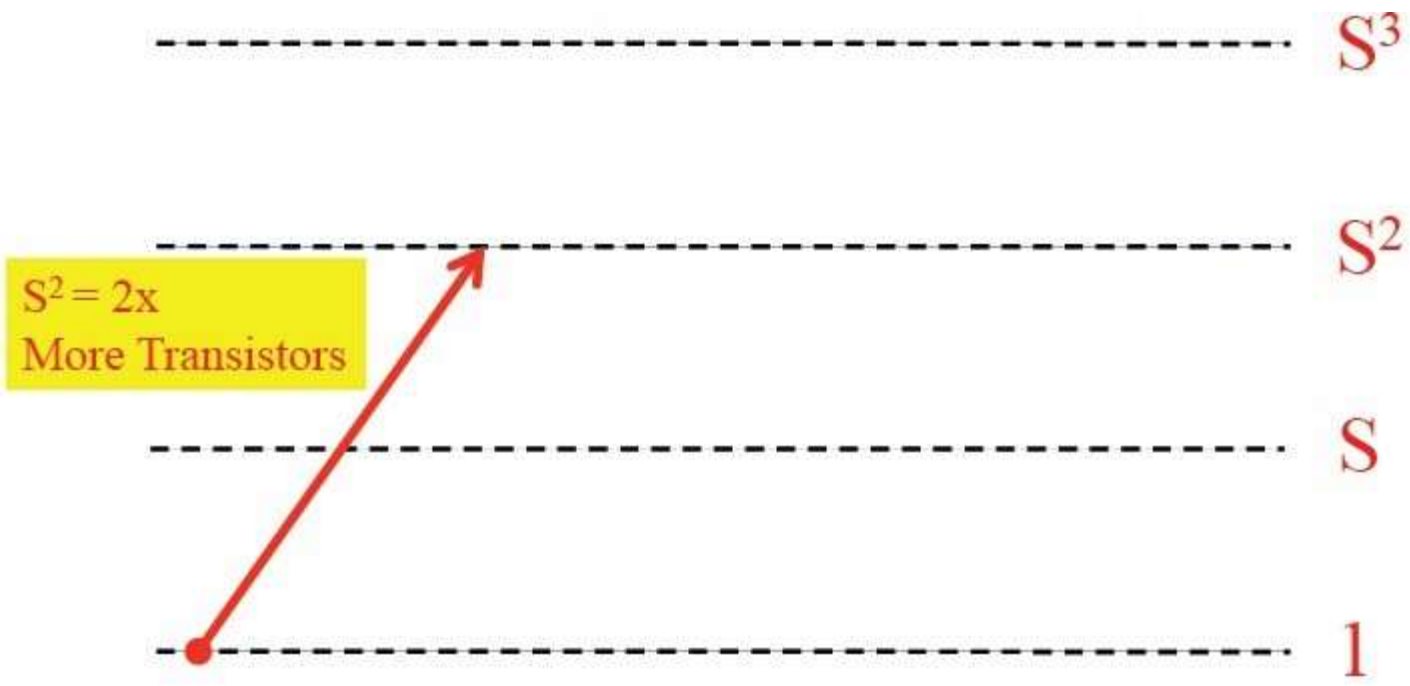
- But, the power budget is constant
  - These gains must be compensated by a corresponding reduction in transistor switching energy

Scaling- why faster transistors?

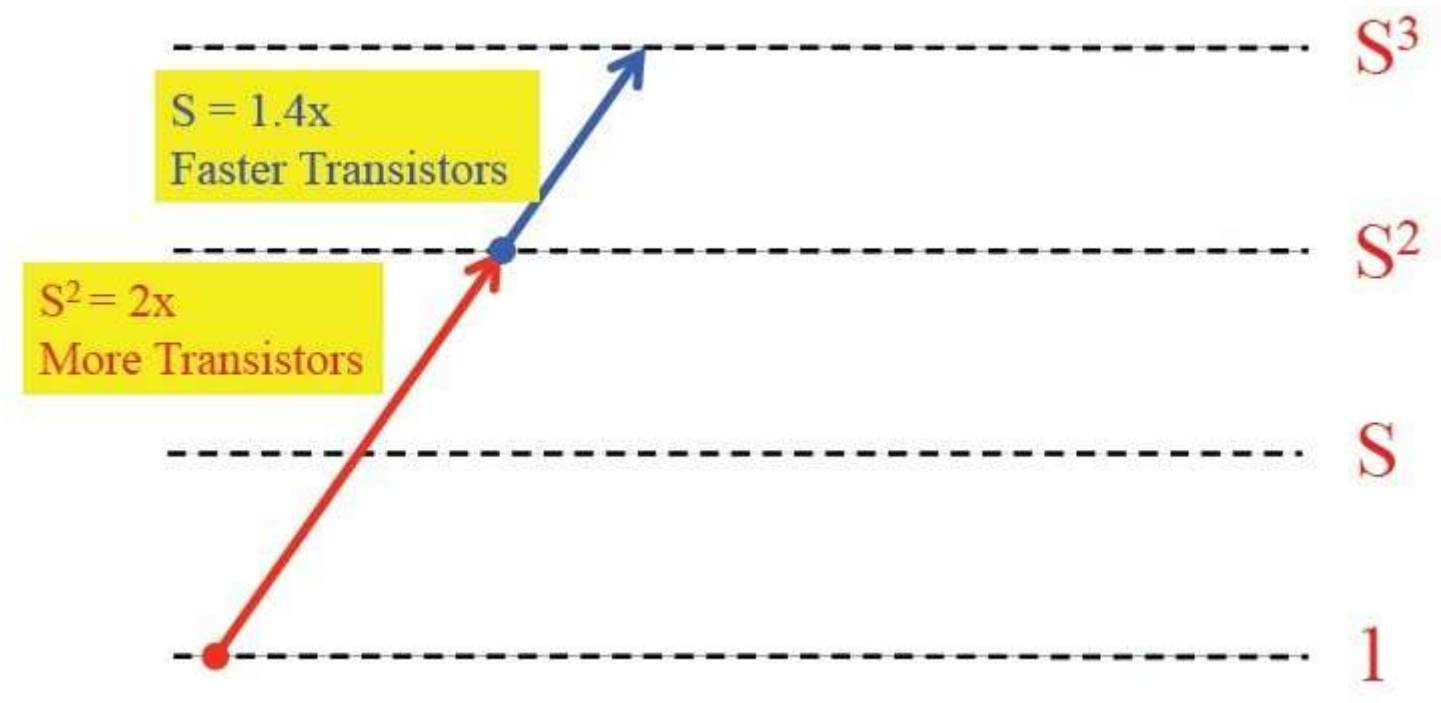
Constant Field Scaling: keep E constant in channel
Constant Voltage Scaling: keep supply voltage constant

Parameters	**Const Field**	**Const Volt**
Dimensions (L and W, Oxide thikness)	$1/\lambda$	$1/\lambda$
Voltage	$1/\lambda$	1
ß ($W/L \times 1/t_{ox}$)	λ	λ^2
R (V/I)	1	$1/\lambda$
C (WL/T_{ox})	$1/\lambda$	$1/\lambda$
Electric Field	1	λ
Current $ß(V-Vt)^2$	$1/\lambda$	λ
Gate Delay (RC)	$1/\lambda$	**$1/\lambda^2$**

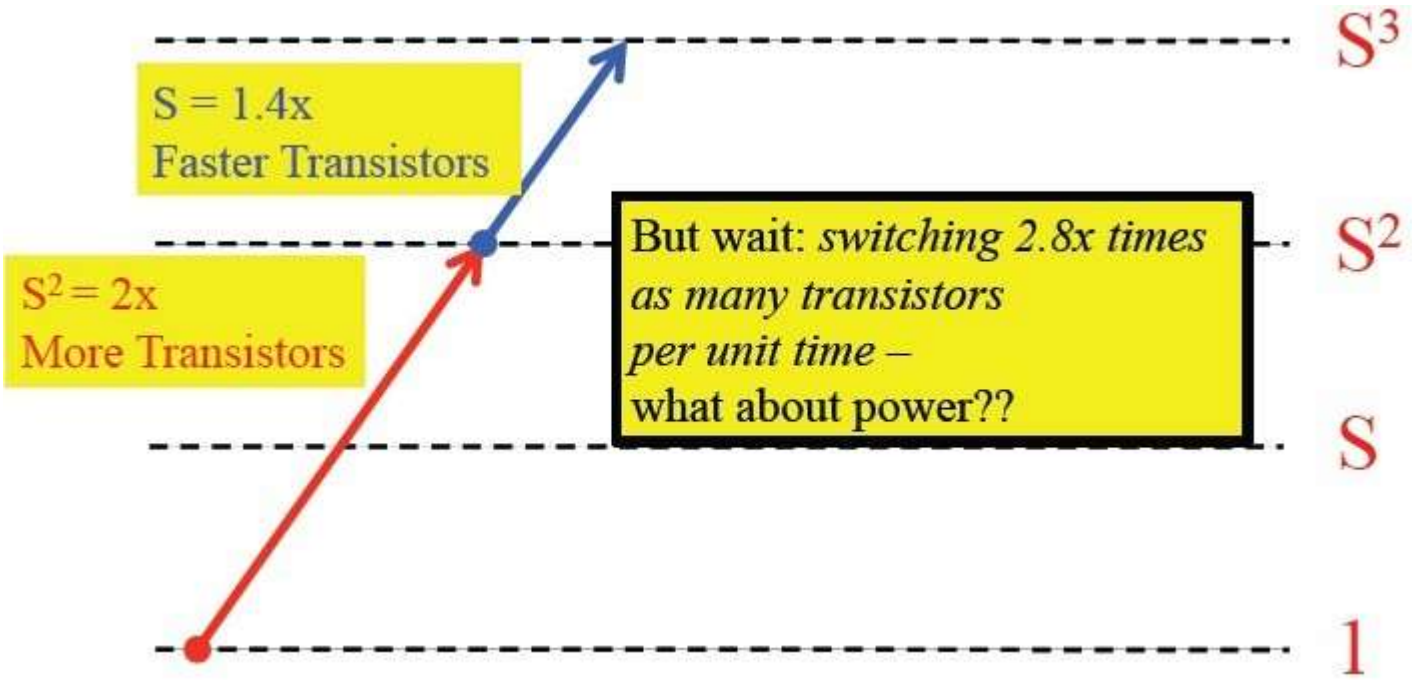
Transistor scaling- power consumption



Transistor scaling

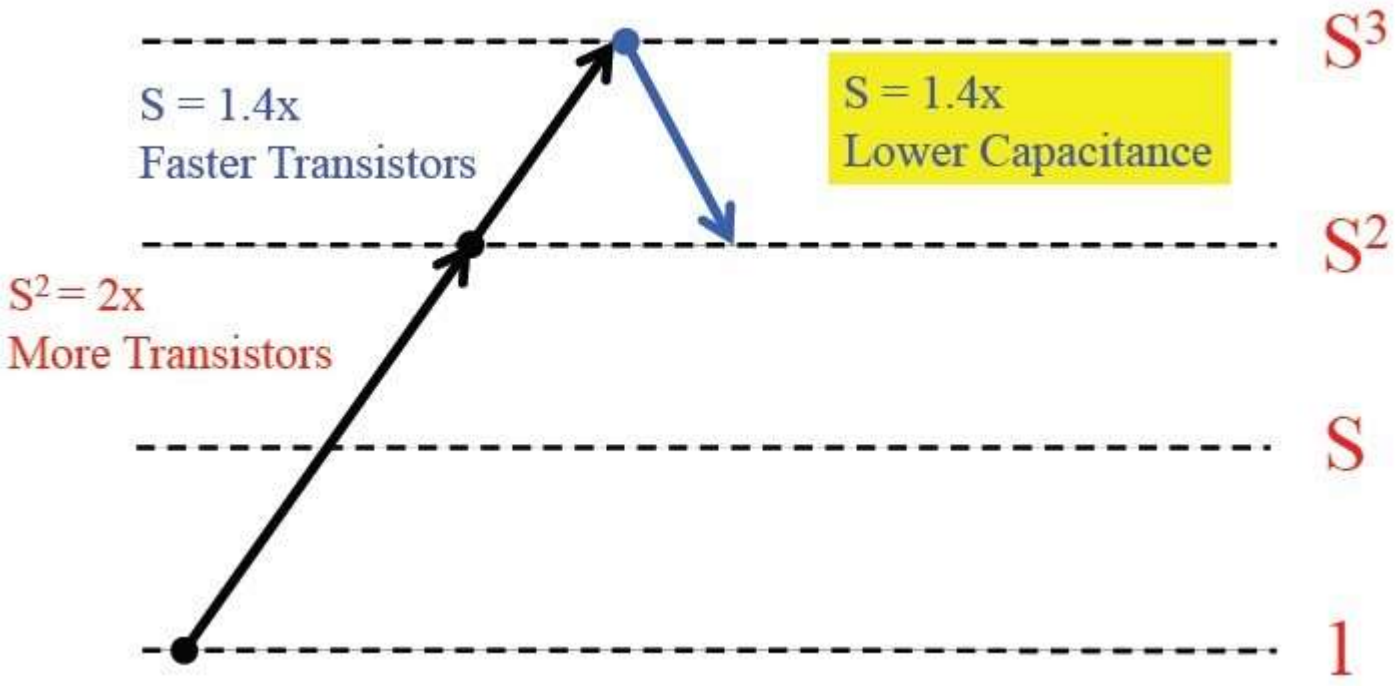


Transistor scaling



$$\text{Power} \approx C\, f\, V_{dd}^2$$

Transistor scaling

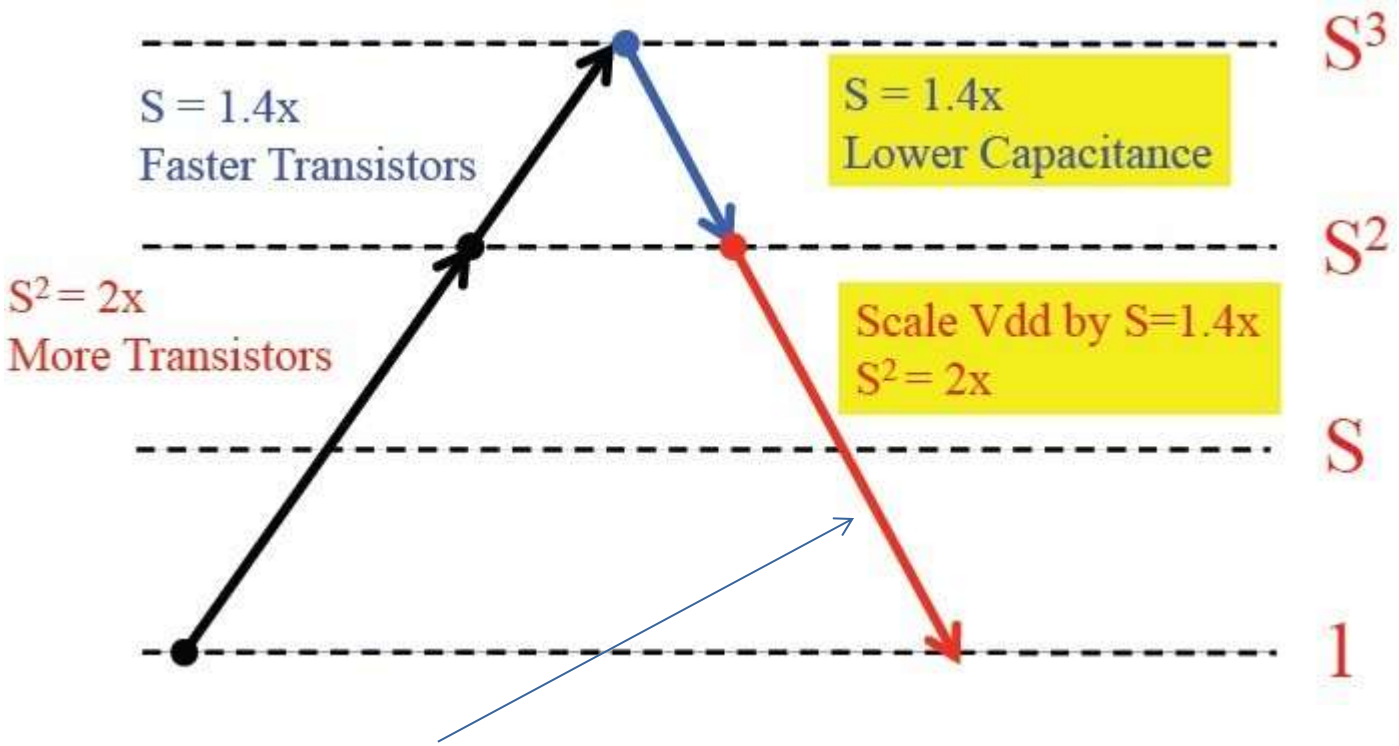


$$\text{Power} \approx C\ f\ V_{dd}^2$$

Transistor scaling

$Power \propto V^2 f$

Voltage Frequency



Dennard Scaling: scale down Vdd and Vth to keep power consumption constant

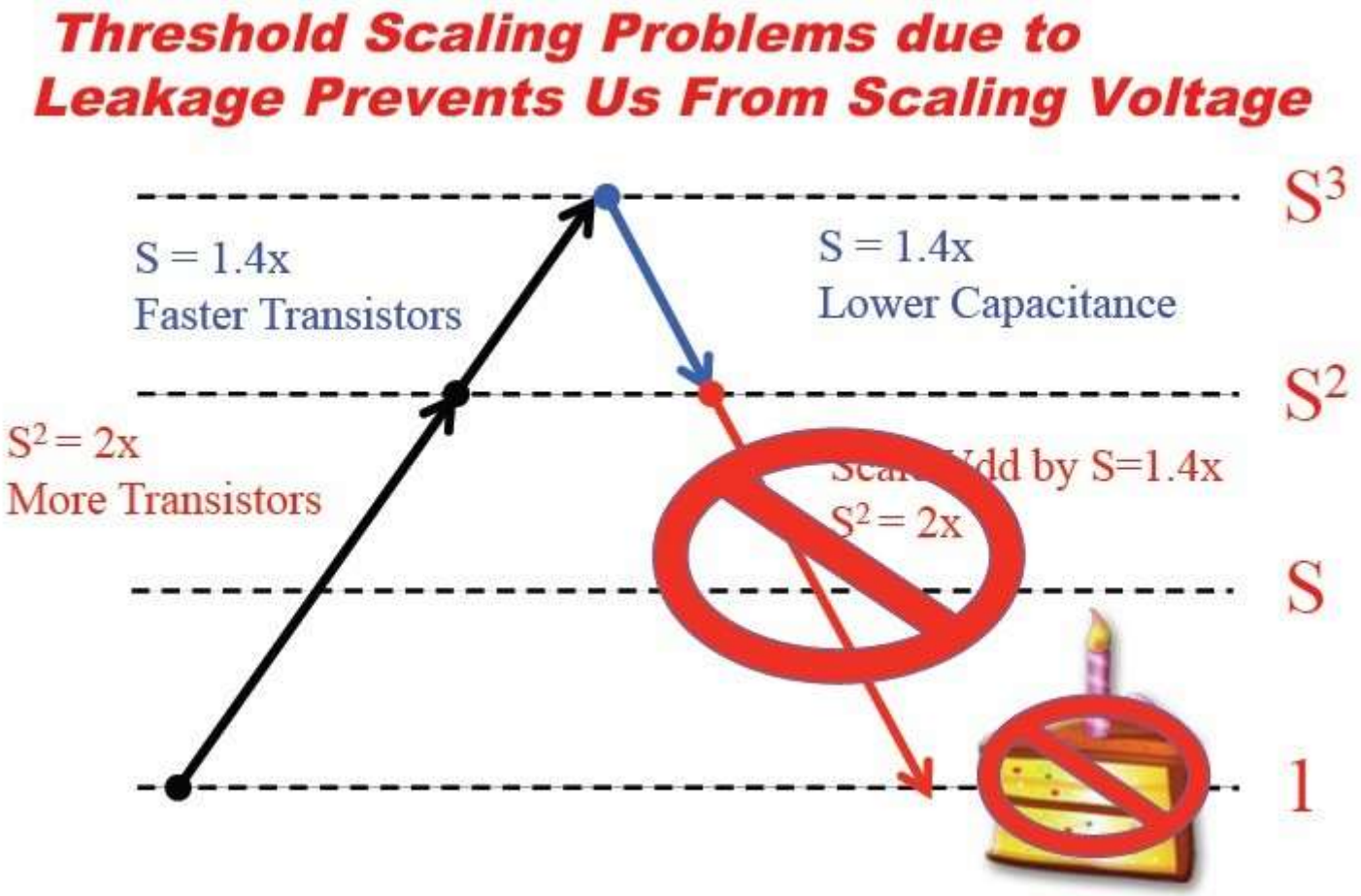
Dennard's scaling rule

Device/Circuit Parameter	Scaling Factor*
Device dimension/thickness	$1/\lambda$
Doping Concentration	λ
Voltage	$1/\lambda$
Current	$1/\lambda$
Capacitance	$1/\lambda$
Delay time	$1/\lambda$
Transistor power	$1/\lambda^2$
Power density	1

IEEE Journal of Solid-State Circuits, Vol. SC-9, October 1974, pp. 256-268.

- By scaling voltage performance goes better with no extra power

Transistor scaling



- After 130 nm technology point,Vdd can hardly be scaled below 1V

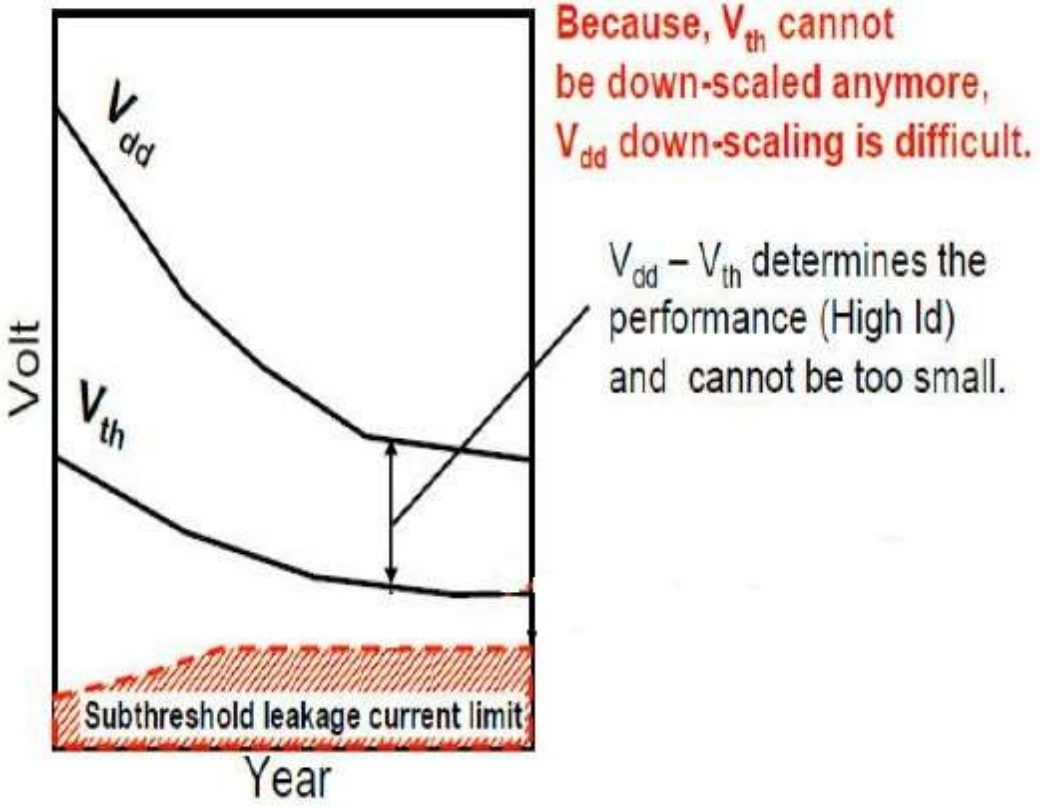
V_{dd} scaling problem

- Bad news:Voltage has always shrunk more slowly (~1/ λ)
- Even worse:Voltage essentially stopped shrinking 10 years ago
  - After 90nm technology point (at 2005),Vdd can hardly be scaled below 1V
  - Reducing voltage increases Sub-threshold leakage current

$$I_{\text{leakage}} \propto \exp\left(-V_{\text{th}}/35mV\right)$$

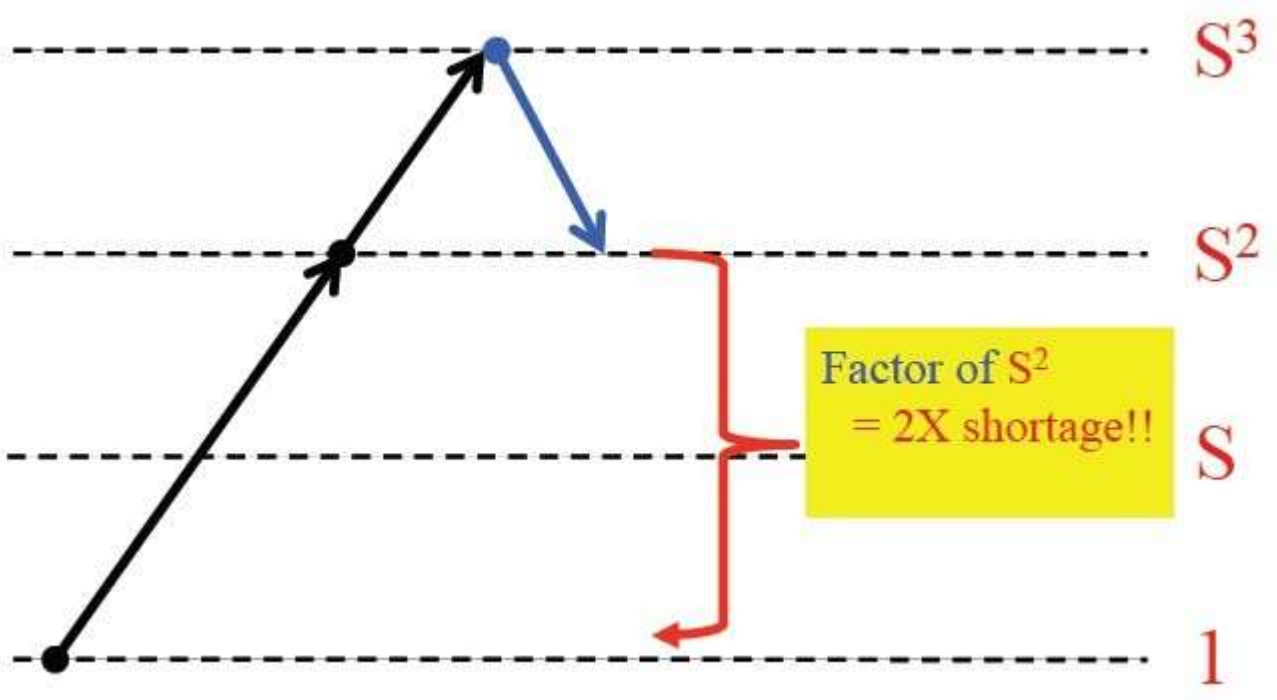
- We entered *Post-Dennard* scaling era

Vdd scaling problem



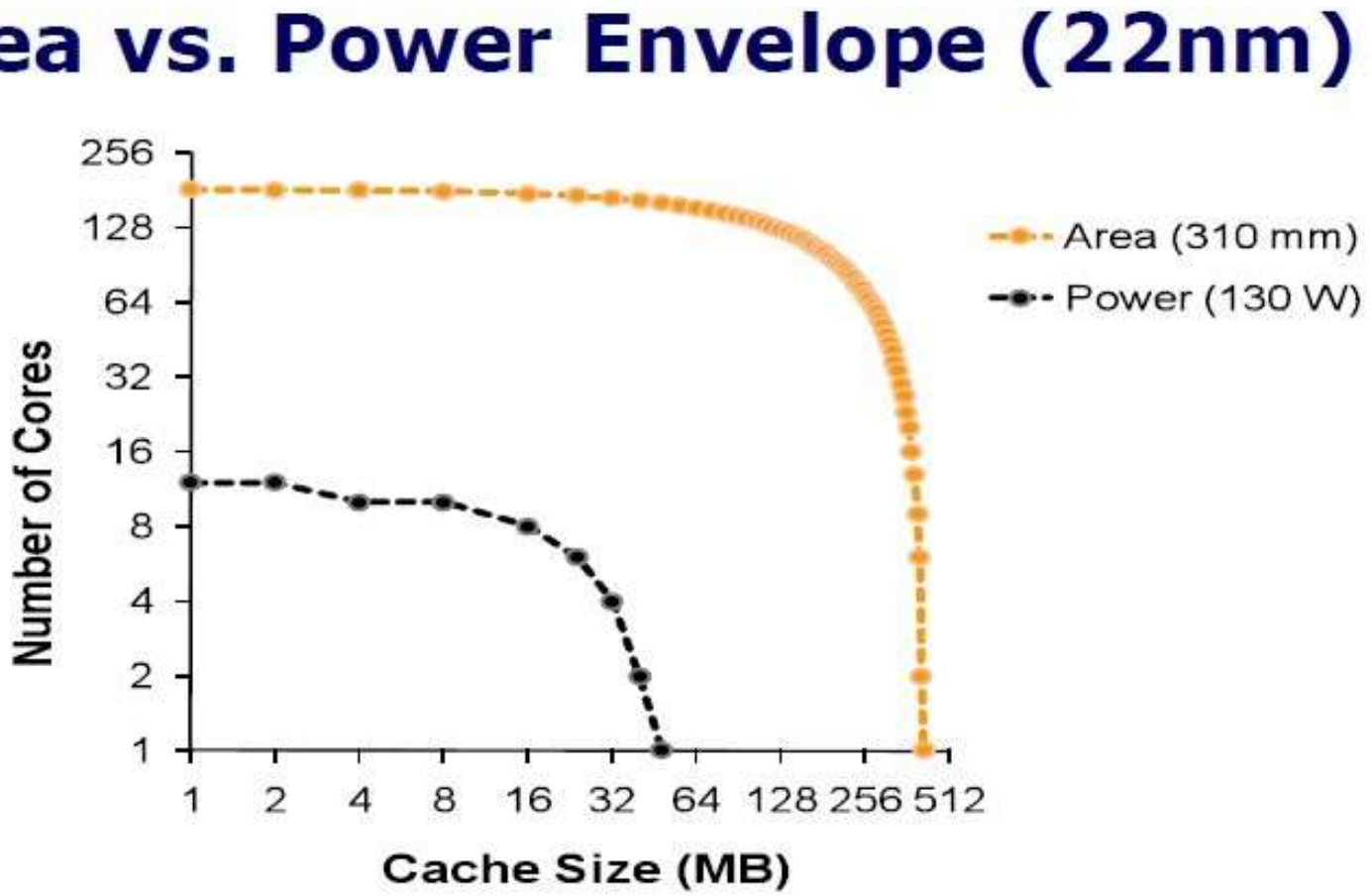
Source: Hiroshi Iwai,Tokyo Institute of Technology

Scaling problem



Full Chip, Full Frequency Power Dissipation Is increasing exponentially by S^2 with every process generation

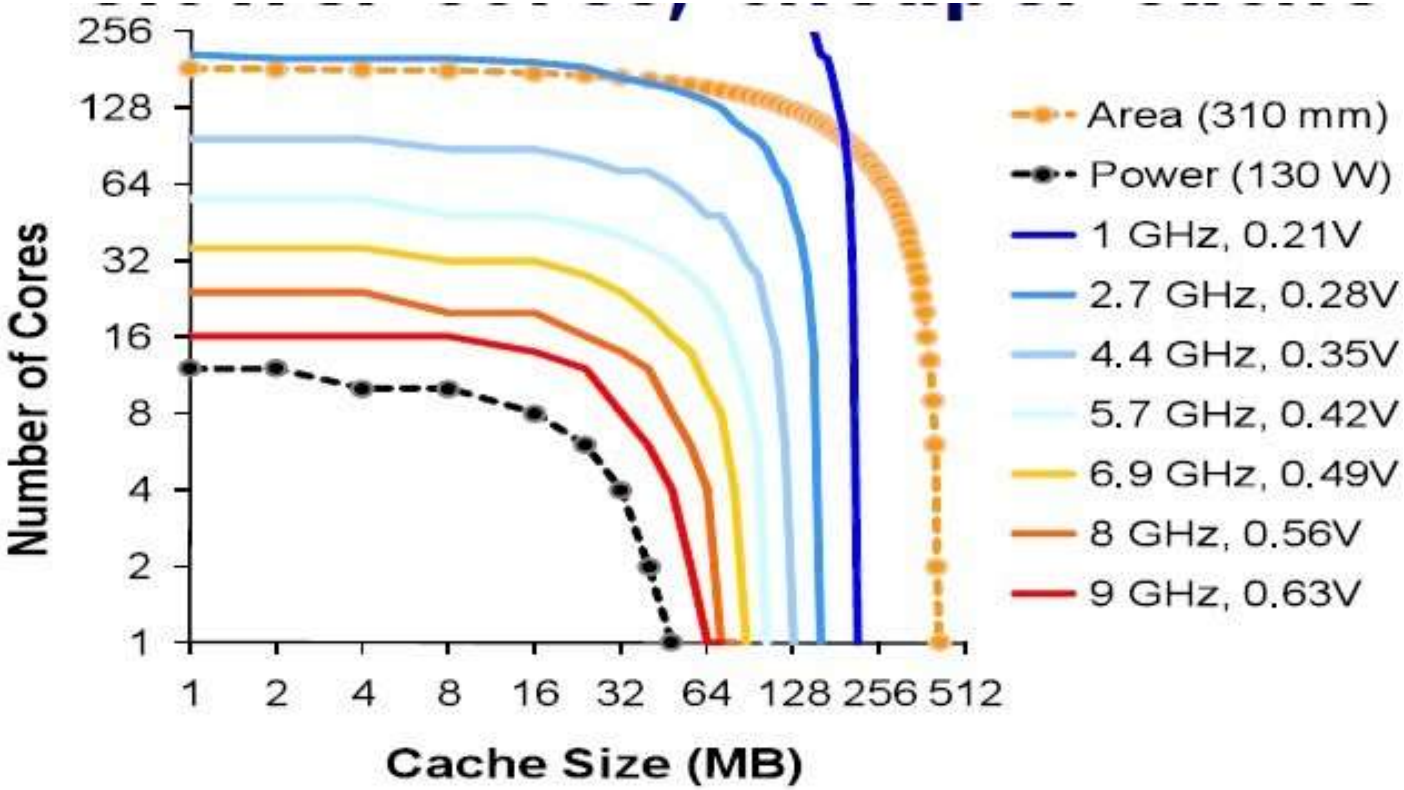
Another view: power wall



✓ Good news: can fit hundreds of cores

× Can not use them all at highest speed

Using frequency scaling

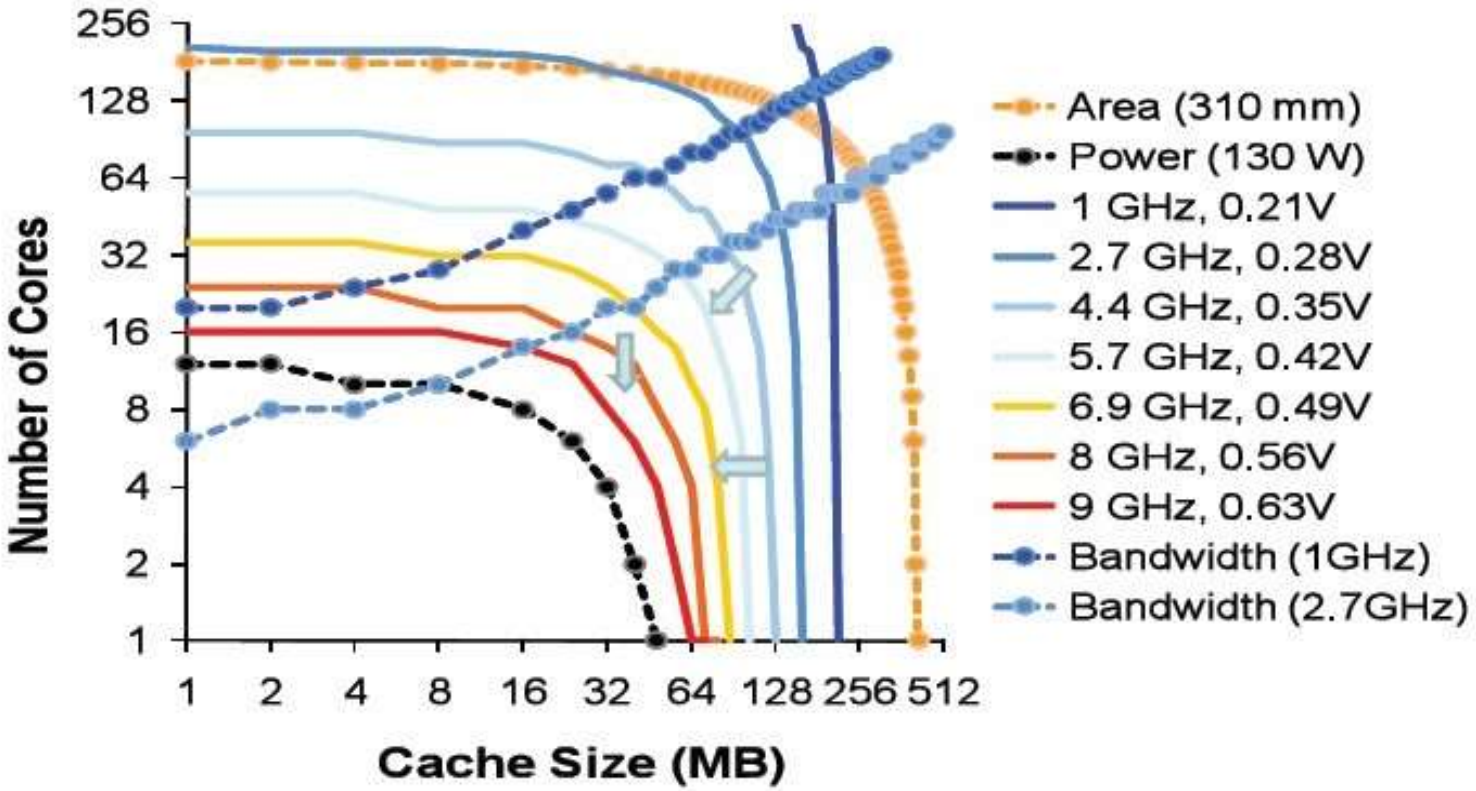


- Result: a performance/power trade-off
- Assuming bandwidth is unlimited

Bandwidth wall

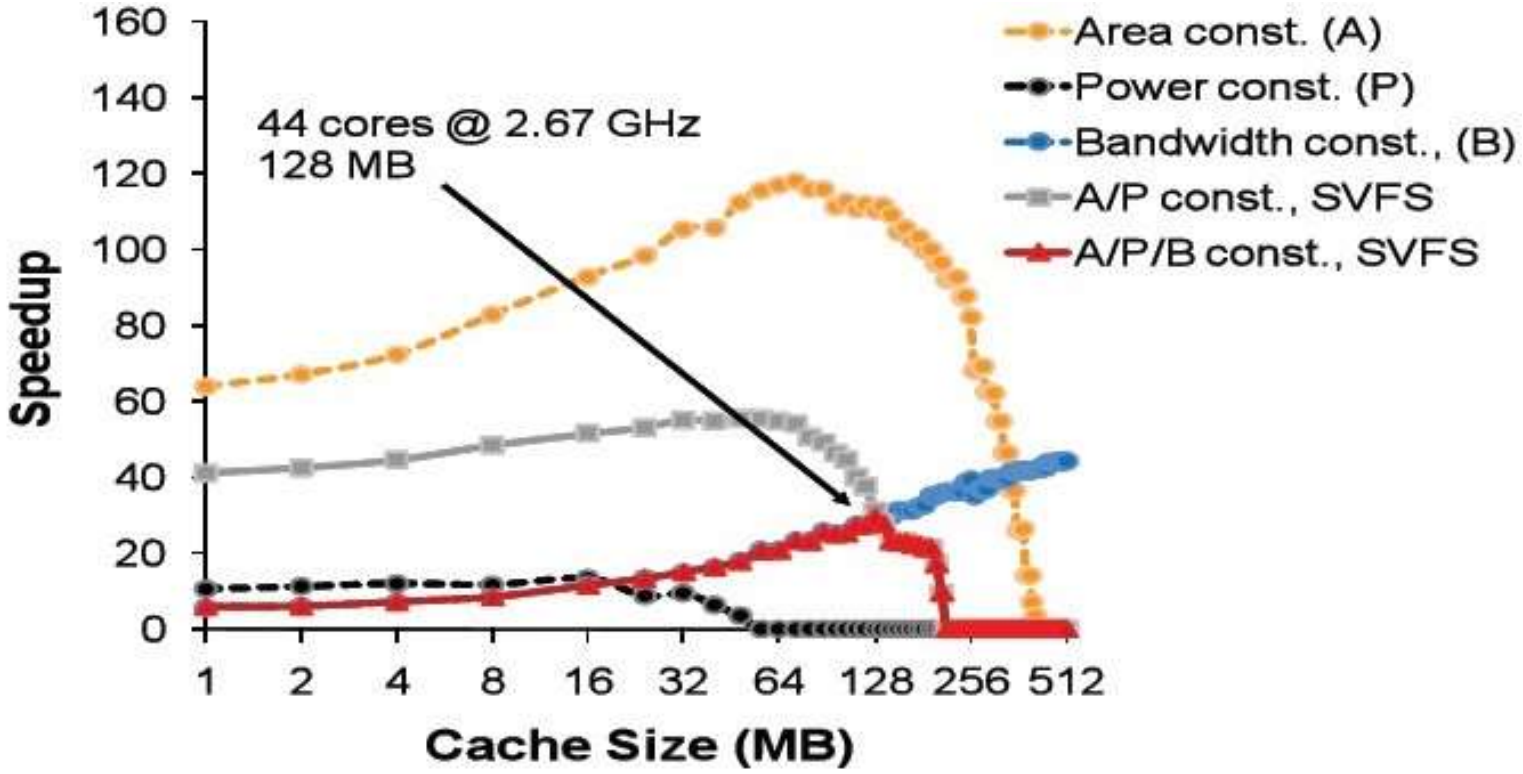
- Another source of underutilization: non-scalable off-chip bandwidth
- Based on the ITRS and other technology projection reports, two-thirds of the pads in a typical many-core processor are used for power and ground
  - Leaving at most one-third available for off-chip memory and I/O signalling.

Bandwidth wall



- For clarity, only showing two bandwidth lines

Speedup vs cache size

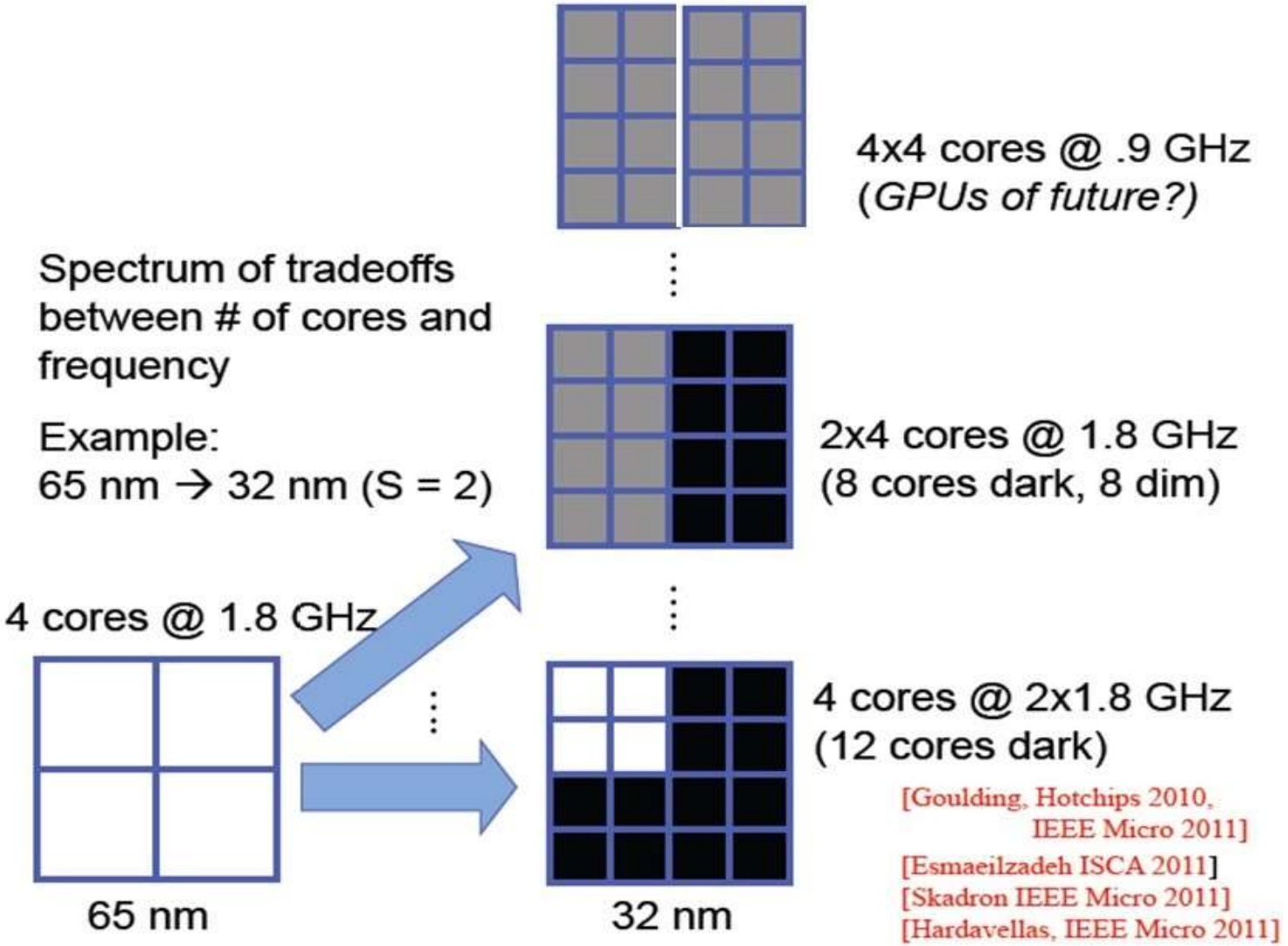


- B/W constrained, then power constrained

Dark silicon

- We can integrate billions of transistors into a single chip, but cannot use all of them due to the power and bandwidth walls
- How can we tackle this problem?
  - Dim silicon
    - Use cores “underclock” to meet the power budget
  - Dark silicon
    - Turn off some parts of the chip to meet the power budget, but run the rest with full frequency

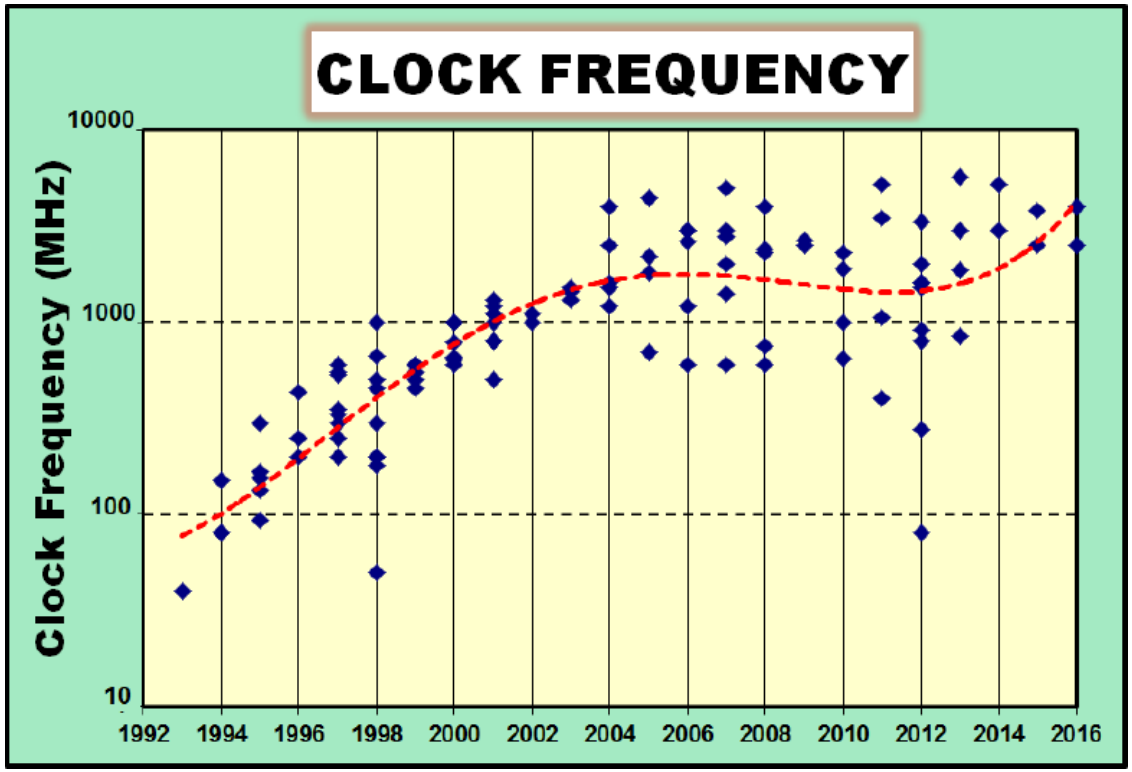
Scaling problem



[Goulding, Hotchips 2010, IEEE Micro 2011]
[Esmaeilzadeh ISCA 2011]
[Skadron IEEE Micro 2011]
[Hardavellas, IEEE Micro 2011]

Dim silicon approach

- Frequency scaling is saturated at around 4GHz
- To stay within TDP



ISSCC 2016

Dim cores

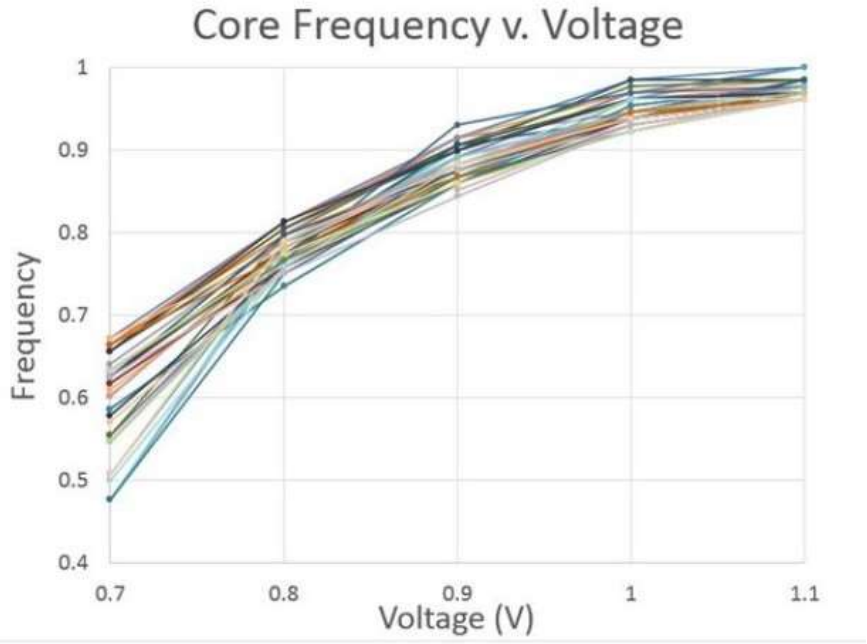
- Dim silicon is the option selected by the industry
- Maximum CPU frequency is limited to 3-4 GHz, but CPUs can work by far faster
- By overclocking, frequency can reach up to 9 GHz
  - Requires using special cooling system and sometimes turning off some cores
- :

Source: hwbot.org

Dim cores

- Modern dim silicon approaches is used in recent processors to save power or give excessive performance
  - Example: AMD TurboCore
    - Used in modern FX processor family

- Another example: IBM Power8 architecture that supports per-core DVFS control with 69 distinct frequency levels



AMD *Turbo Core* technology

- Performance boosting technology: automatically switches from six cores to three turbocharged core
- For applications that just need speed over a few cores
- In Turbo CORE mode, the AMD Phenom™ II shifts frequency speed from 3.2GHz on six cores, to 3.6GHz on three cores
  - Based on utilization level: 6 levels
- Doesn't shut down the three unused cores; it set them down to 800 MHz and drops their voltage
  - keeps the processor within its TDP

Dark Silicon

- Many solutions based on dark silicon
  - Specialized cores to tackle power wall
  - Emerging memory technologies to tackle bandwidth wall

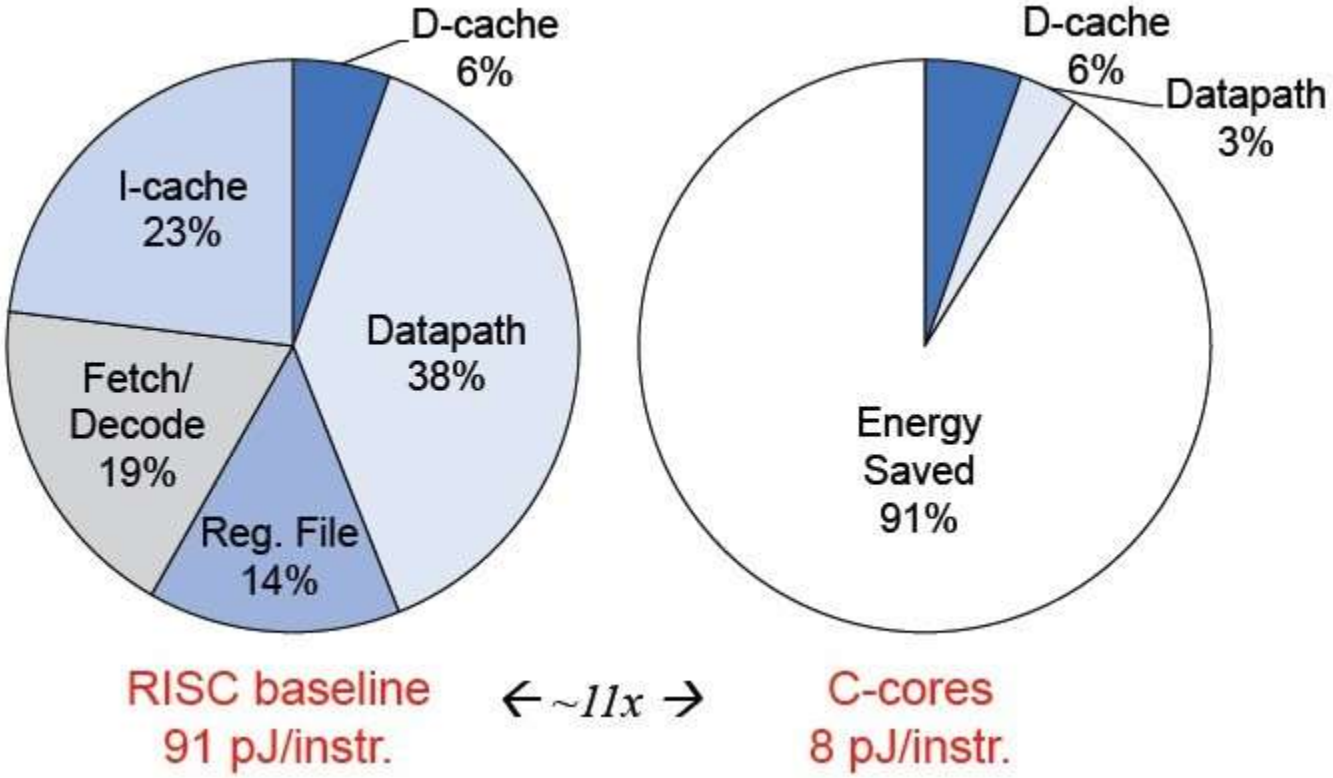
Question?

Specialized cores

- Trade dark silicon for power-efficiency
  - Use all of that dark silicon area to build specialized cores, each of them tuned for the task at hand (10-100x more energy efficient), and only turn on the ones we need
  - Activating only those cores that are most useful to the running application while leaving the vast majority of the on-chip cores powered down
  - Use inexpensive area to save expensive energy

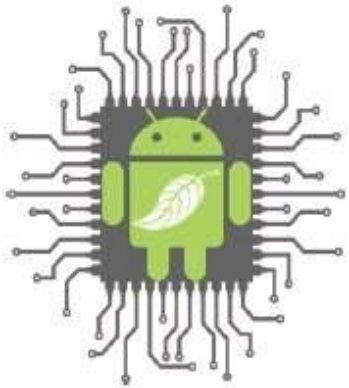
Specialized cores

- Hardware implementation is more power-efficient than software implementation



Specialized cores

- The first commercial platform that exploits dark silicon is the ARM big.LITTLE platform
- Some more complex systems in academia
  - Find specialized cores based on the target application of the SoC and build a many-core SoC by these cores
  - Example: GreenDroid by UCSD

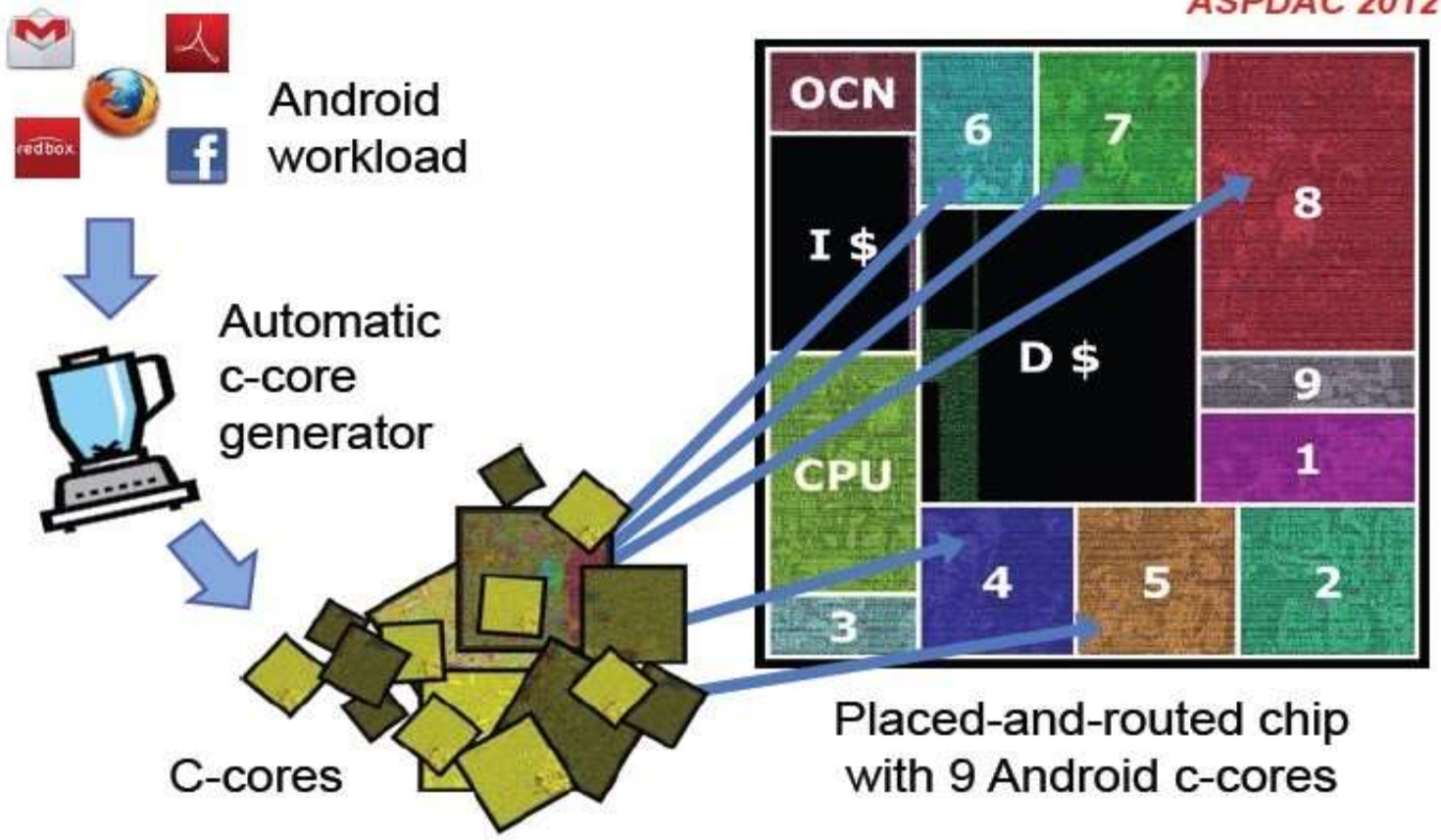


GreenDriod

"GreenDroid: A Mobile Application Processor for a Future of Dark Silicon" *HOTCHIPS AUG 2010*

IEEE Micro Mar 2011

ASPDAC 2012

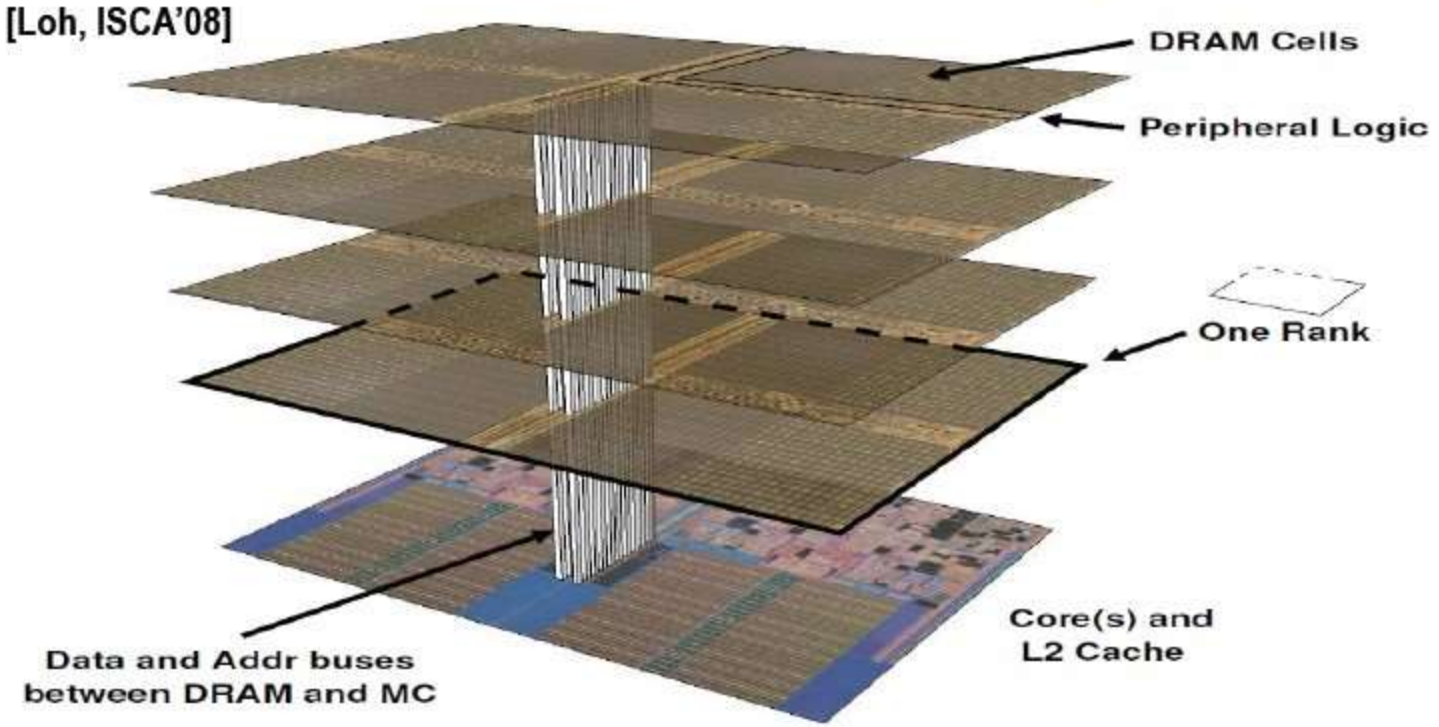


Bandwidth wall

- Specializing brings about a great power-efficiency
- How to push the bandwidth wall?
  - 3D memories: building memories on top of the CPU on the same chip

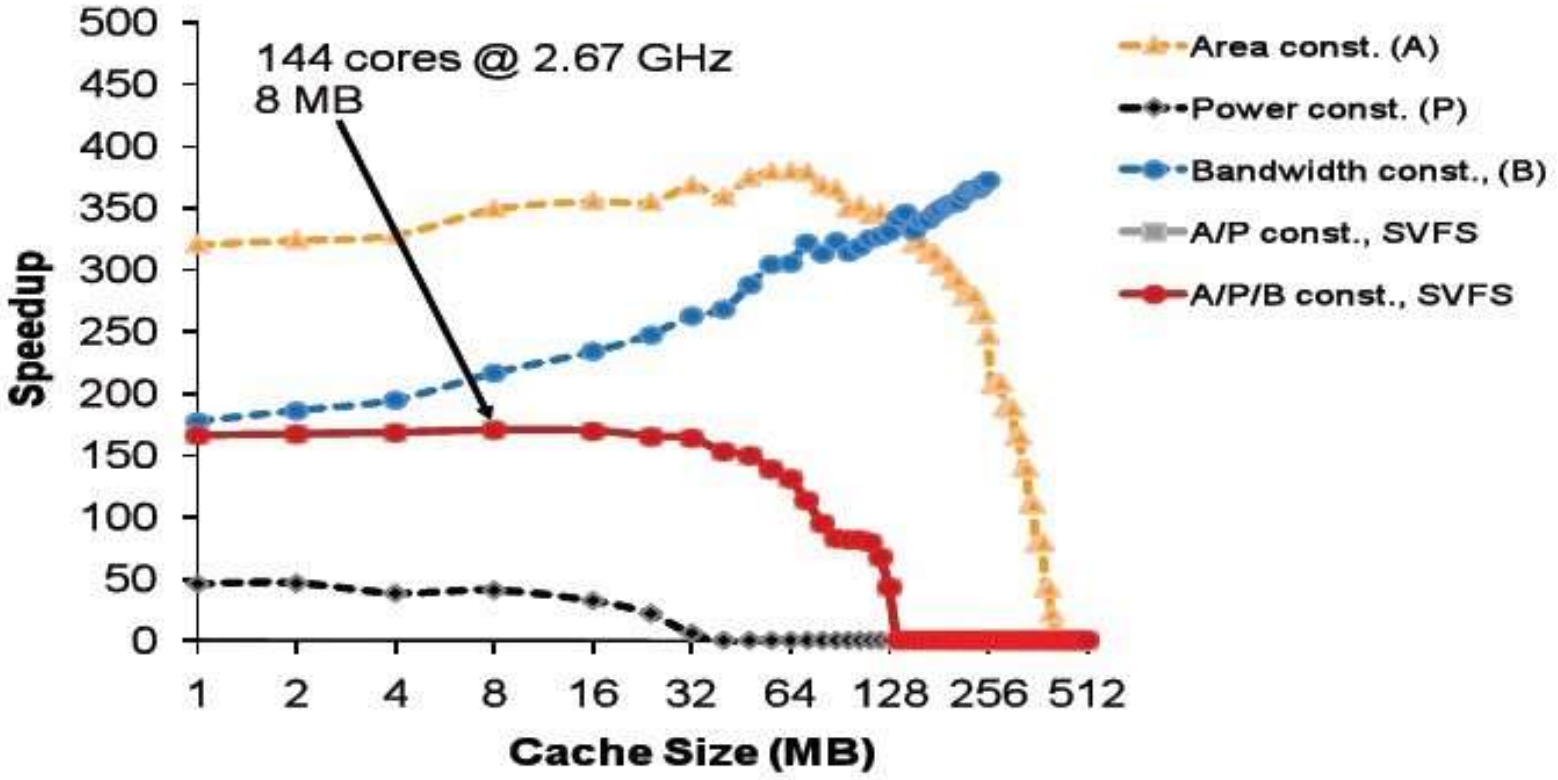
3D memory

Mitigating B/W Limitations: 3D-stacked Memory



- Delivers TB/sec of bandwidth

Peak Performing with 3D Memory



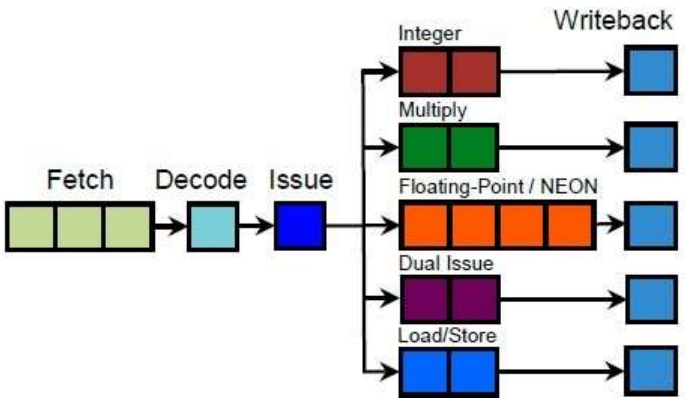
- Only power-constrained

big.LITTLE platform

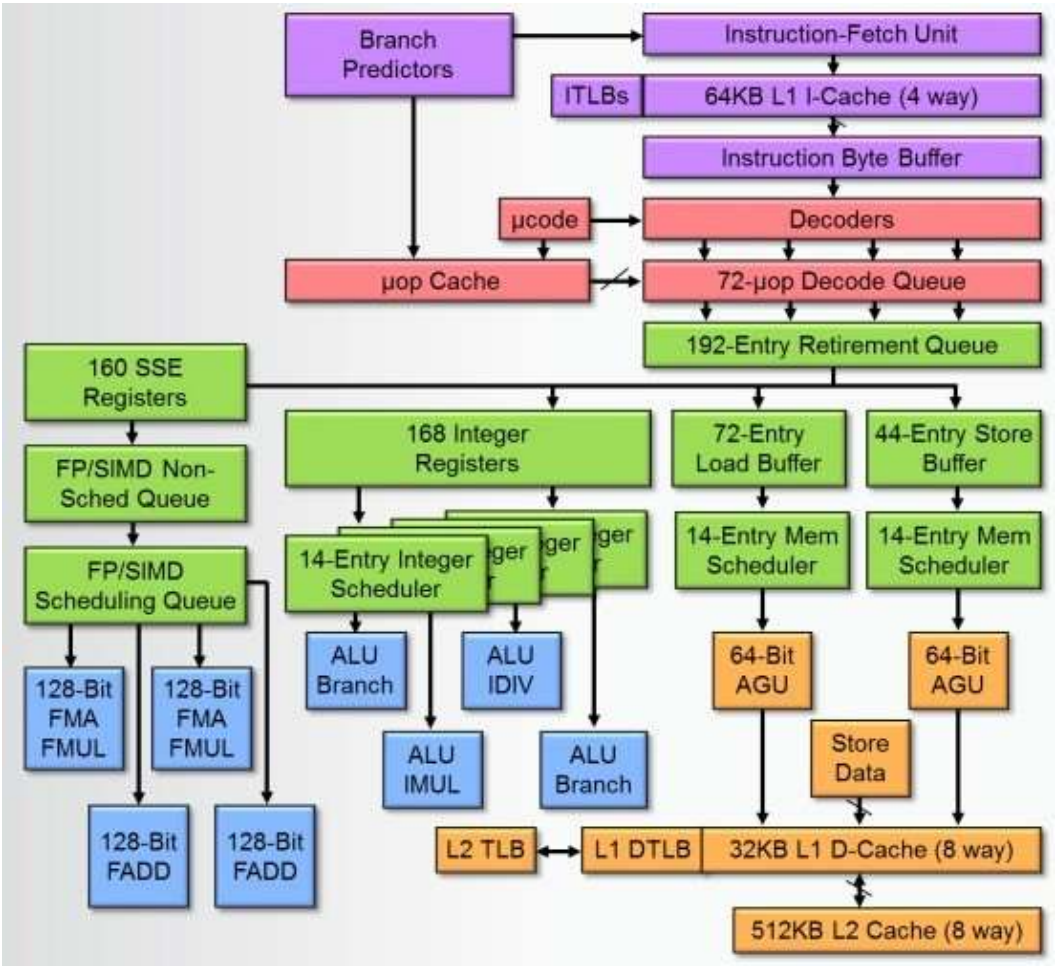
- Different operational points got by DVFS

Core complexity and power usage

- Simple processors often use lower power to perform a given task (e.g. an multiply operation) than their complex (and fast) counterparts



ARM A7 pipeline (source: arm.com)



AMD Zen pipeline (source: microprocessor report)

big.LITTLE platform

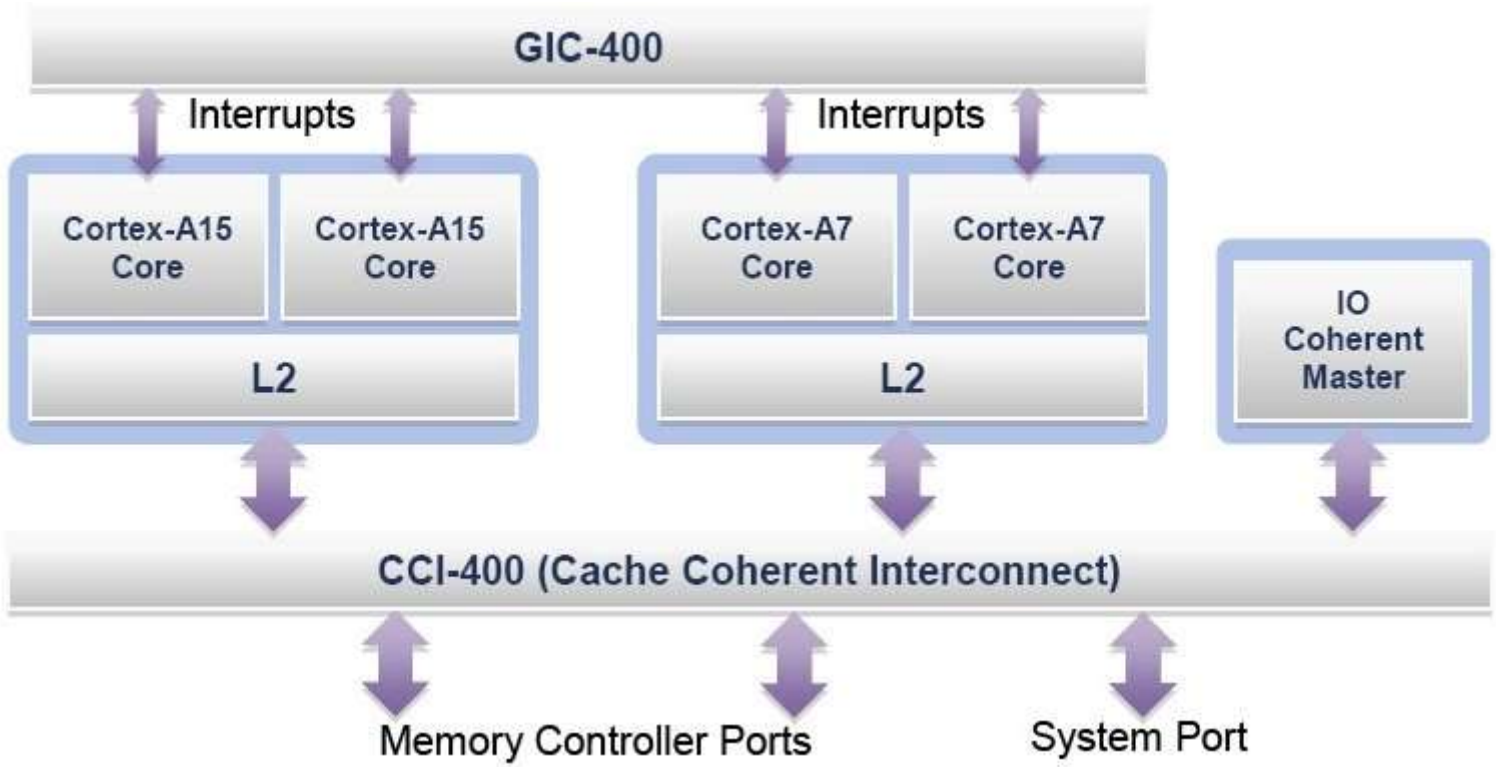
- Basic idea: use low-power little cores to save power, when user does not care about performance, but use big cores when performance is needed
- The difference in performance and energy between Cortex-A15 and Cortex-A7 across a variety of benchmarks and micro-benchmarks

	Cortex-A15 vs Cortex-A7 Performance	Cortex-A7 vs Cortex-A15 Energy Efficiency
Dhrystone	1.9x	3.5x
FDCT	2.3x	3.8x
IMDCT	3.0x	3.0x
MemCopy L1	1.9x	2.3x
MemCopy L2	1.9x	3.4x

big.LITTLE platform

- A ‘big’ ARM® Cortex™-A15 processor is paired with a ‘LITTLE’ Cortex™-A7 processor to create a system that can accomplish both high intensity and low intensity tasks in the most energy efficient manner
- The processors are architecturally identical
  - Both Cortex-A15 and Cortex-A7 implement the full ARM v7A architecture

System diagram

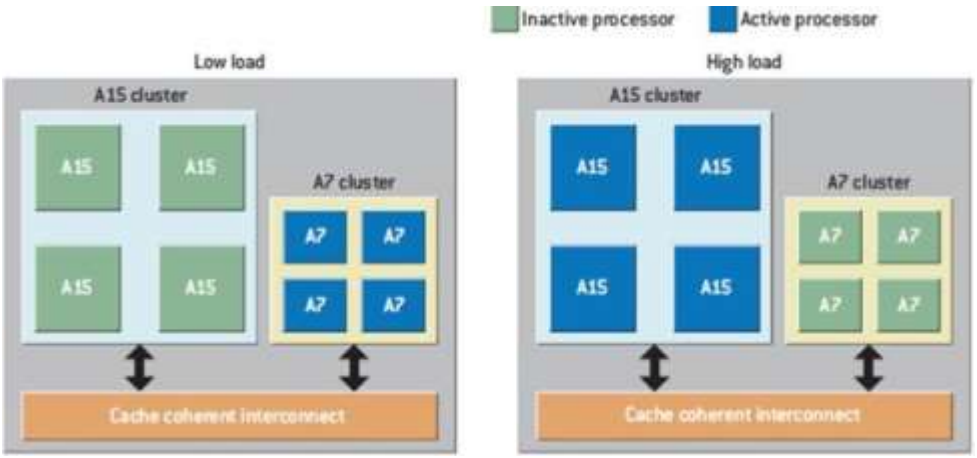


big.LITTLE use models

- *Two Use models*: the policy based on which the processors are used
- There are three models that can be used for big.LITTLE processing:
  - Cluster migration
  - CPU migration
  - Heterogeneous Multi-Processing model

Cluster migration model

- The cores are grouped into two "big" and "LITTLE" cores
- The operating system scheduler can only see one cluster at a time
  - When the load on the whole processor changes between low and high, the system transitions to the other cluster.
  - All data and working set is passed through a common L2 cache
- Implemented in the Samsung Exynos 5 Octa, and nVidia Tegra X1

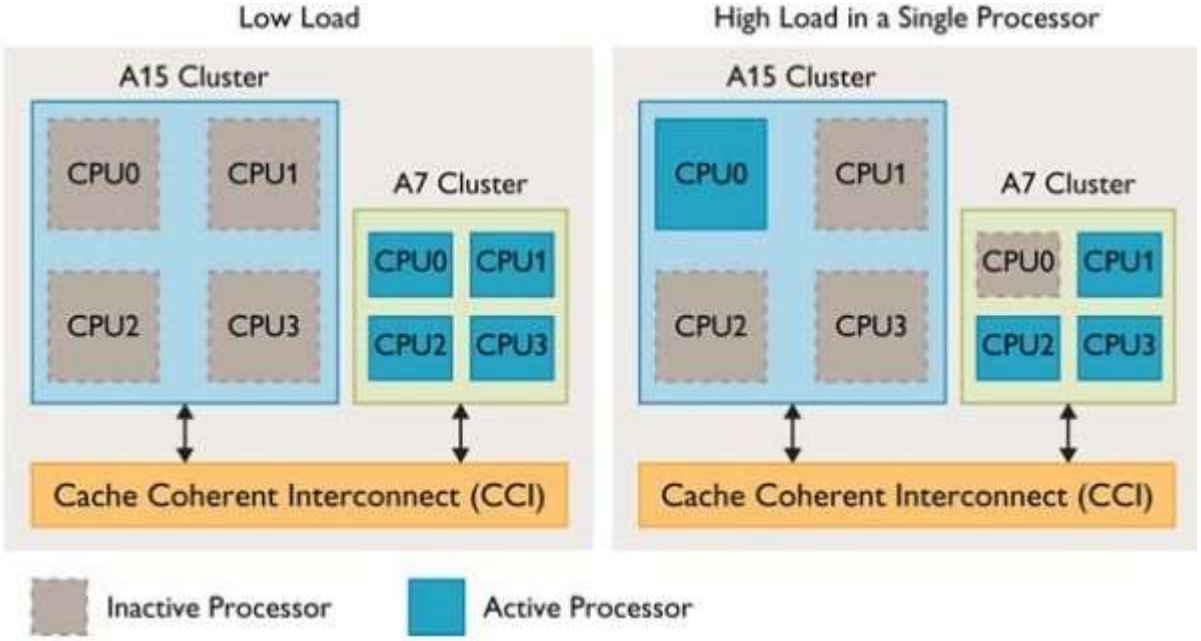


CPU migration model

- The big.LITTLE cpu migration use model
  - Only Cortex-A15 or Cortex-A7 and never both processors are active at the same time
- Once Cortex-A7 is at its highest operating point if more performance is required a task migration can be invoked that picks up the OS and applications and moves them to Cortex-A15
- the Cortex-A15-Cortex-A7 system is designed to migrate in less than 20,000-cycles

CPU migration model

- Each core in big cpu is paired with a core in LITTLE cpu
  - Core 0 on the Cortex-A15 and Cortex-A7 processors, core1 on the Cortex-A15 and Cortex-A7 processors and so on
- Migration occurs between paired cores
- This model allows a mix of big and LITTLE cores to be active at any one time



Heterogeneous MP use model

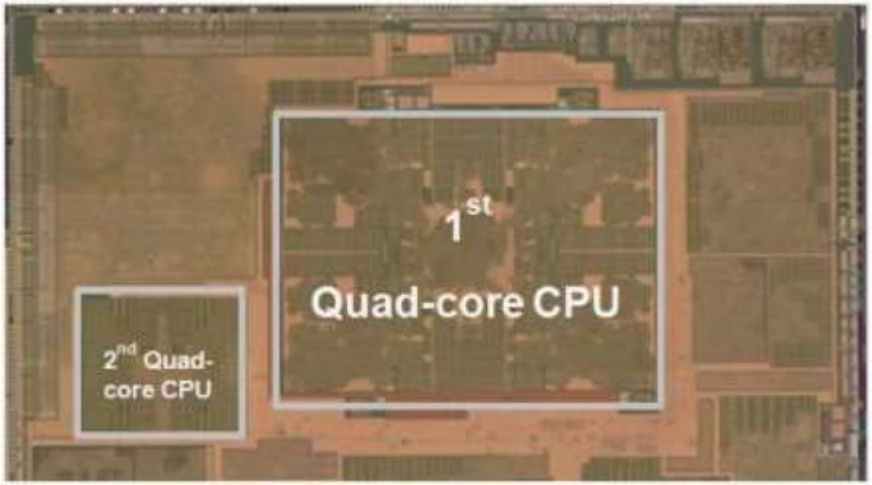
- Both Cortex-A15 and Cortex-A7 can be powered on and simultaneously executing code
- Cortex-A15 only needs to be powered on and simultaneously executing next to Cortex-A7 if there are threads that need that level of processing performance
  - If not, only Cortex-A7 needs to be powered on
- Compute intensive threads that require significant amounts of processing performance can be allocated to Cortex-A15
- Threads that are I/O heavy or that do not produce a result that is time critical to the user can be executed on Cortex-A7

Samsung Exynos 5 Octa

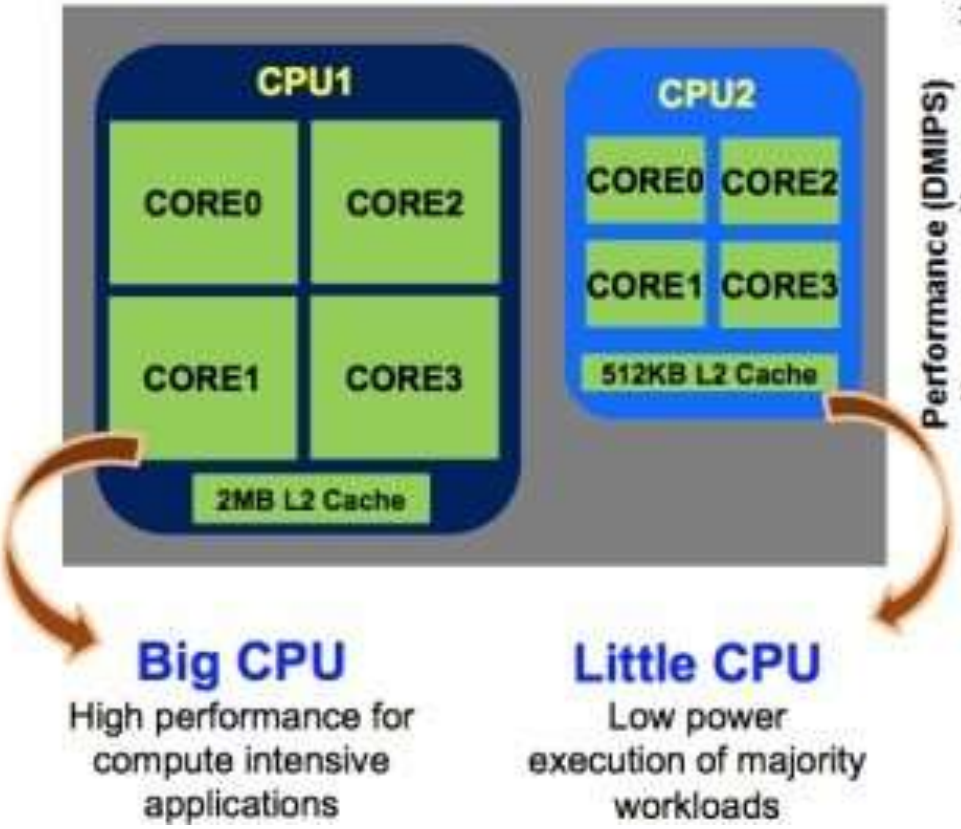
- The first commercial big.LITTLE implementation
- 1.6-1.8 GHz quad-core ARM Cortex-A15 and 1.2 GHz quad-core ARM Cortex-A7
- The CPU emulates a single quad-core processor to the operating system and applications
- Migration model:
  - At any given time, either the big cluster is active (for maximum performance) or the LITTLE cluster (for maximum power efficiency)
  - The switching of the active cluster is performed on demand by the chip firmware

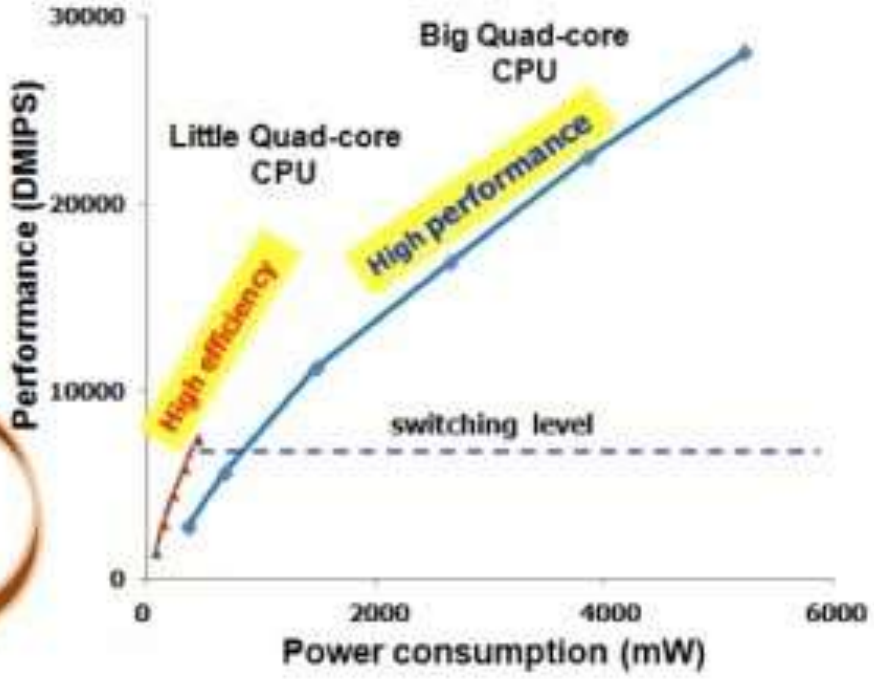
Samsung Exynos 5 Octa

	1st Quad-core CPU	2nd Quad-core CPU
Architecture	ARM v7a	ARM v7a
Process	Samsung 28nm HKMG	Samsung 28nm HKMG
Frequency	200MHz~1.8GHz+	200MHz~1.2GHz+
Area	19mm^2	3.8mm^2
Power-ratio	1	0.17

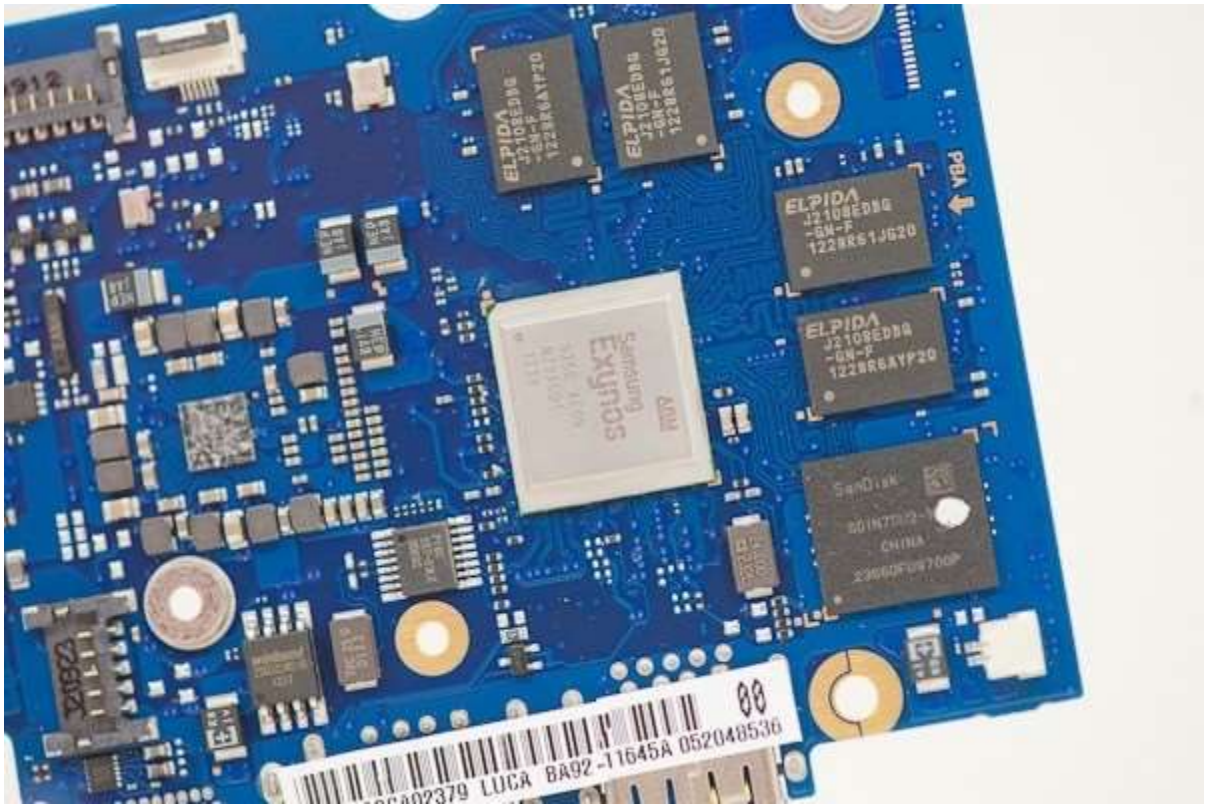


Samsung Exynos 5 Octa



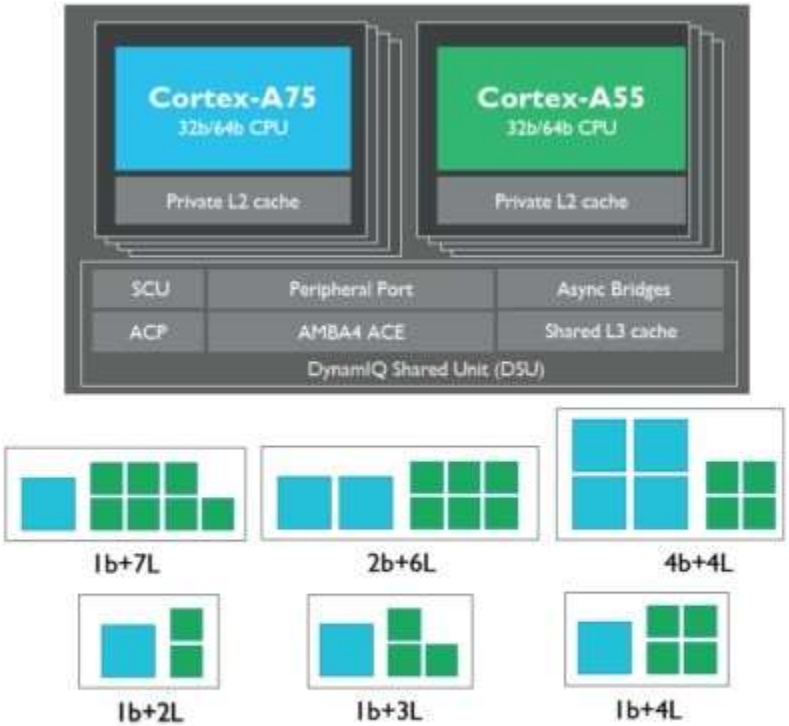


Samsung Exynos 5 Octa board



ARM DynamIQ

- The newest and more flexible big.LITTLE model



Example DynamIQ big.LITTLE configurations

ARM

nVidia 4-plus-1 technology

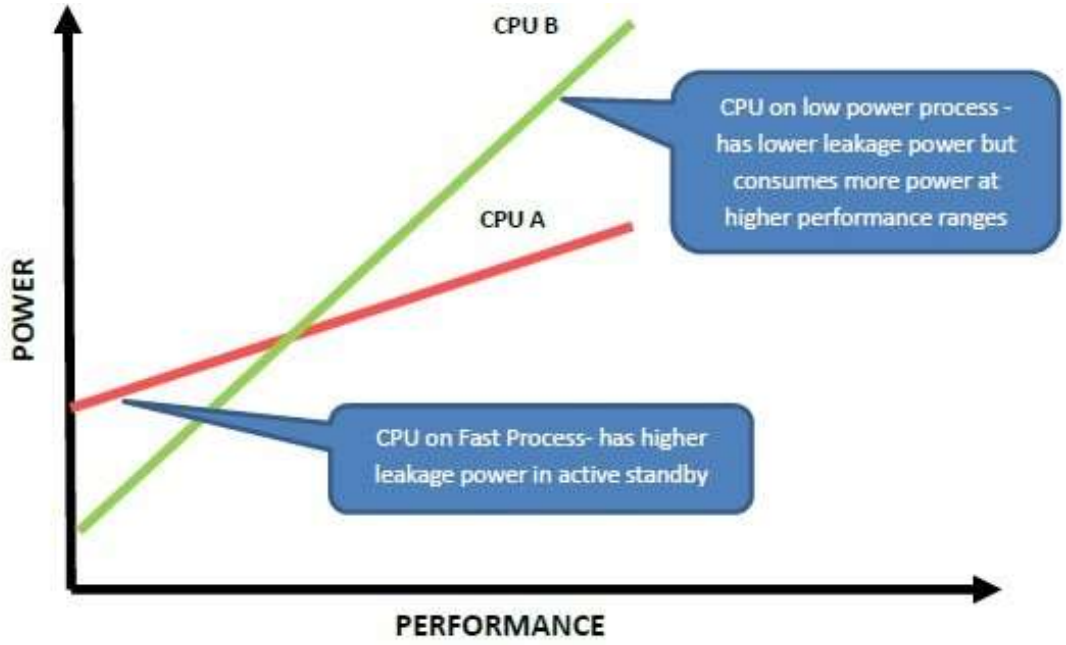
- Like big.LITTLE, introduced by nVidia in 2012
- Also known as *Variable Symmetric Multiprocessing (vSMP)*
- 4+1 architecture consisting of four “big” Cortex cores and one Cortex “battery saver” core
- All five CPU cores are identical
- 1+4 will never be powered on simultaneously
  - At any time, only 1 (the battery saver) or 4 (main cores) are active

nVidia 4-plus-1 technology

- "Battery Saver core" built using a special low power silicon process
  - Executes tasks at low frequency for active standby mode: when the user is not actively interacting with the device
- The four main "quad" cores are built using a standard silicon process to reach higher frequencies
  - Used in active mode: browsing the web, checking Email, gaming, running multimedia applications, playing back media, and so on

nVidia 4-plus-1 technology

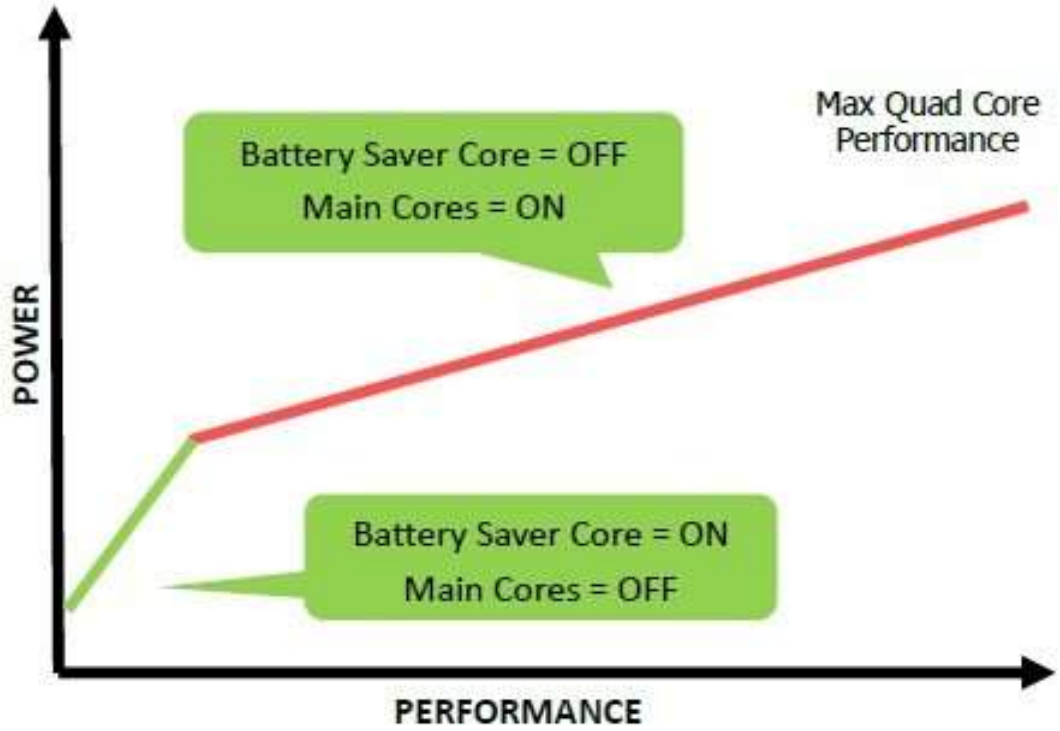
Fast and low process

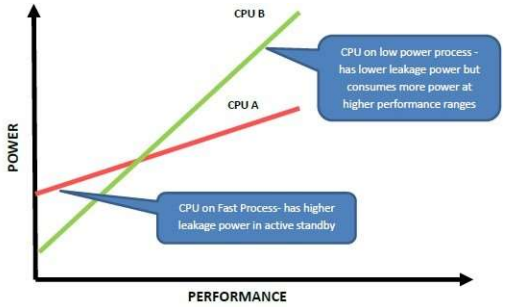


Fast Process = Optimized for high frequency operation, but higher leakage

Low Power Process = Operates at lower frequency with lower leakage

Fast and low process





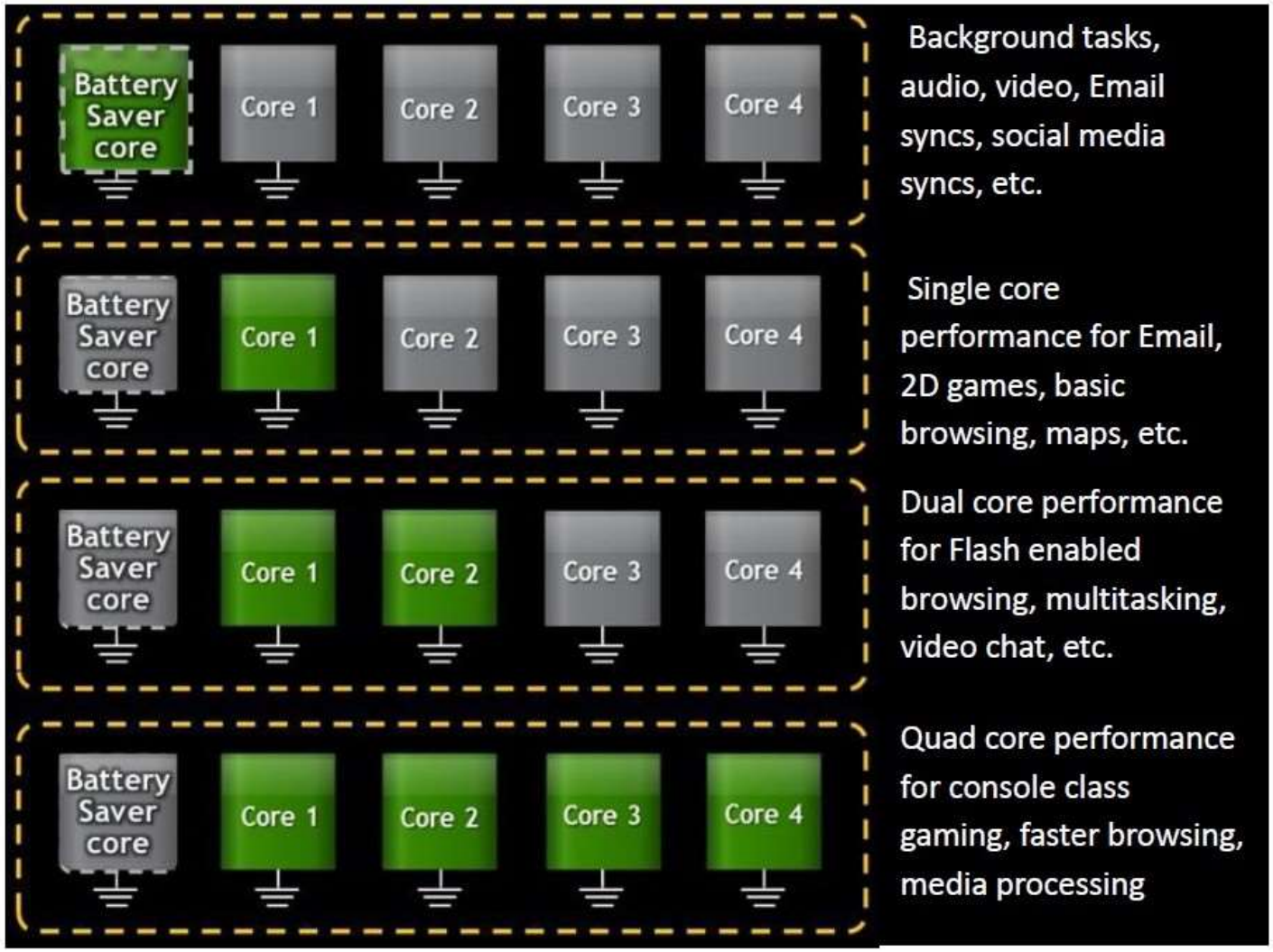
- Ideal: use the cores that best match the current CPU load
  - Combine the advantageous part of both curves

nVidia 4-plus-1 technology

- The “Battery Saver” core is OS transparent
  - OS and applications are not aware of this core, but automatically take advantage of it
  - Unlike current Asynchronous SMP architectures
- A hardware and software CPU management logic continuously monitors CPU workload to automatically and dynamically enable and disable the Battery Saver core and the main CPU cores

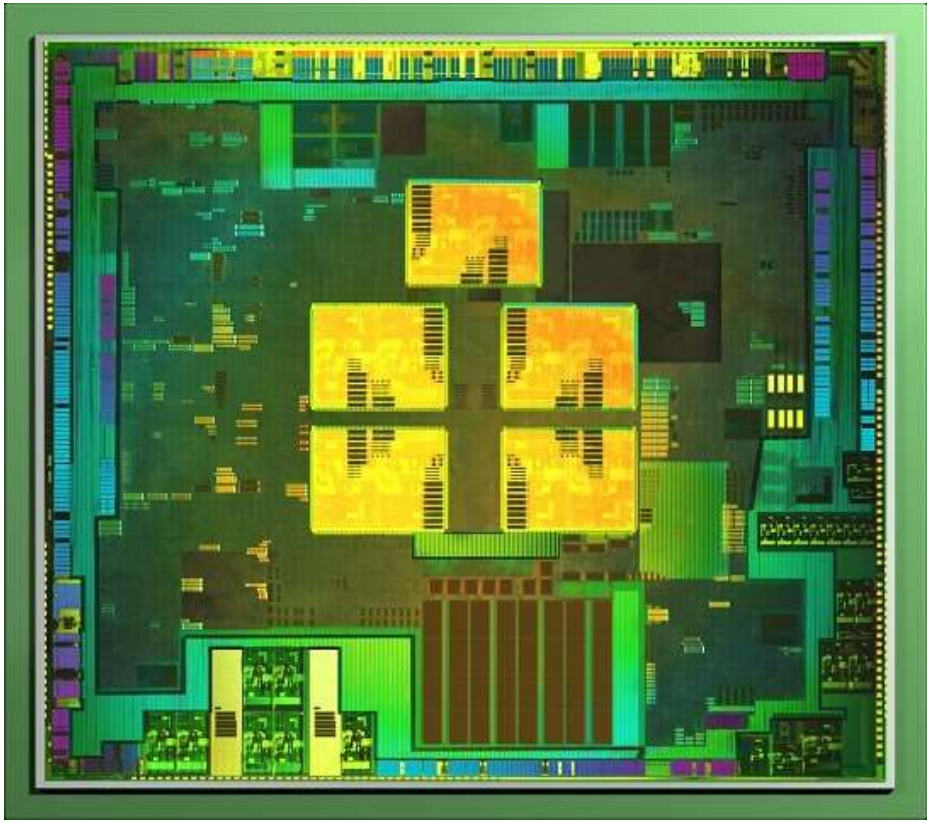
nVidia 4-plus-1 technology

- In addition to selecting between the battery saver and main cores, the number of active cores are also changed dynamically based on CPU load



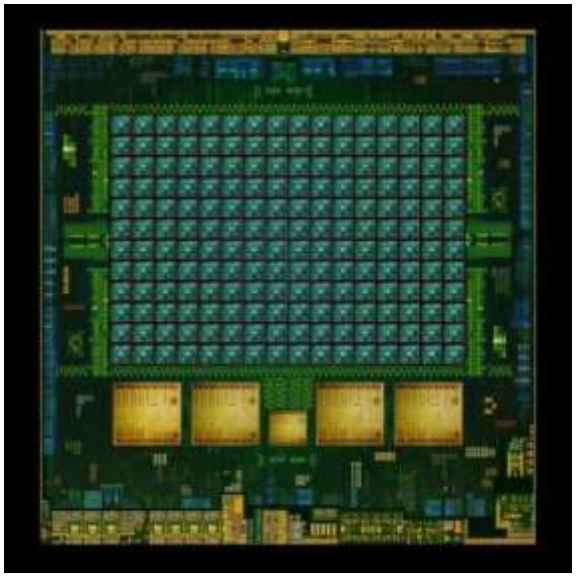
nVidia Tegra 3

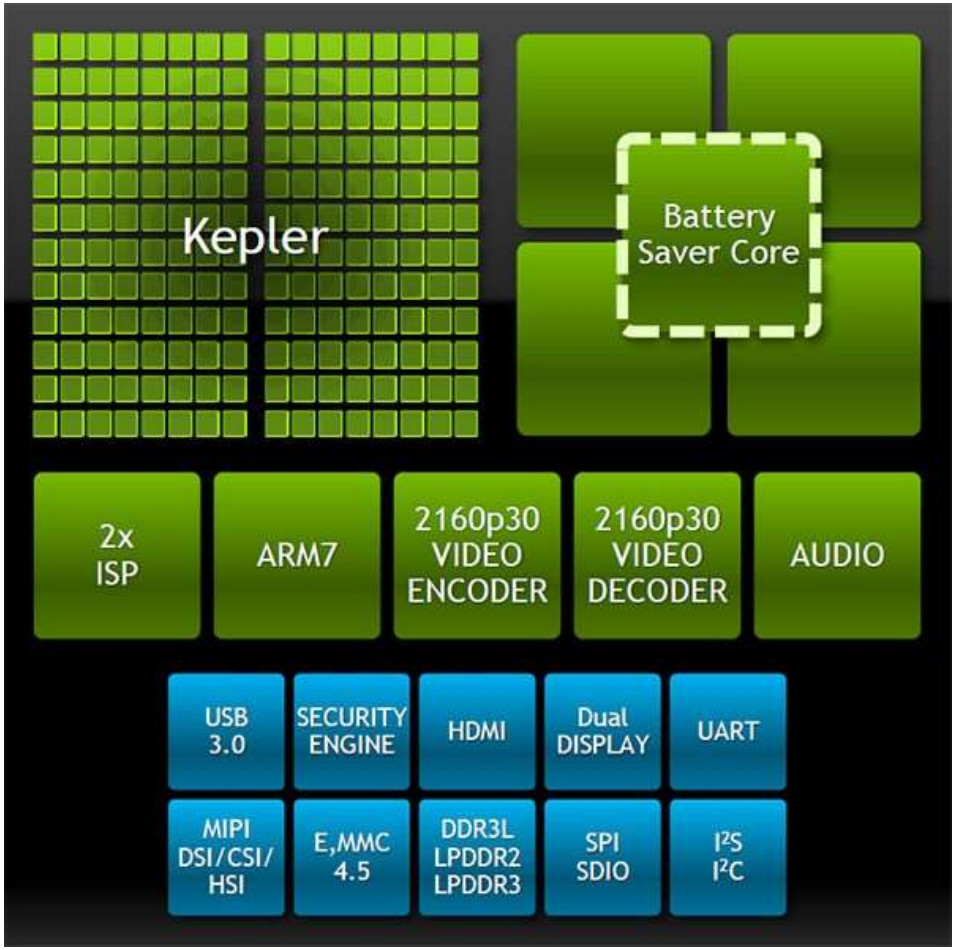
- The first implementation of 1-plus-4 technology
- One slow Cortex A9 optimized for low power, and 4 fast Cortex A9 optimized for performance.The slow one will run at 500 MHz while the fast ones could be 1 GHz or more
- Used in many devices, e.g. HTC One X



Tegra-K1

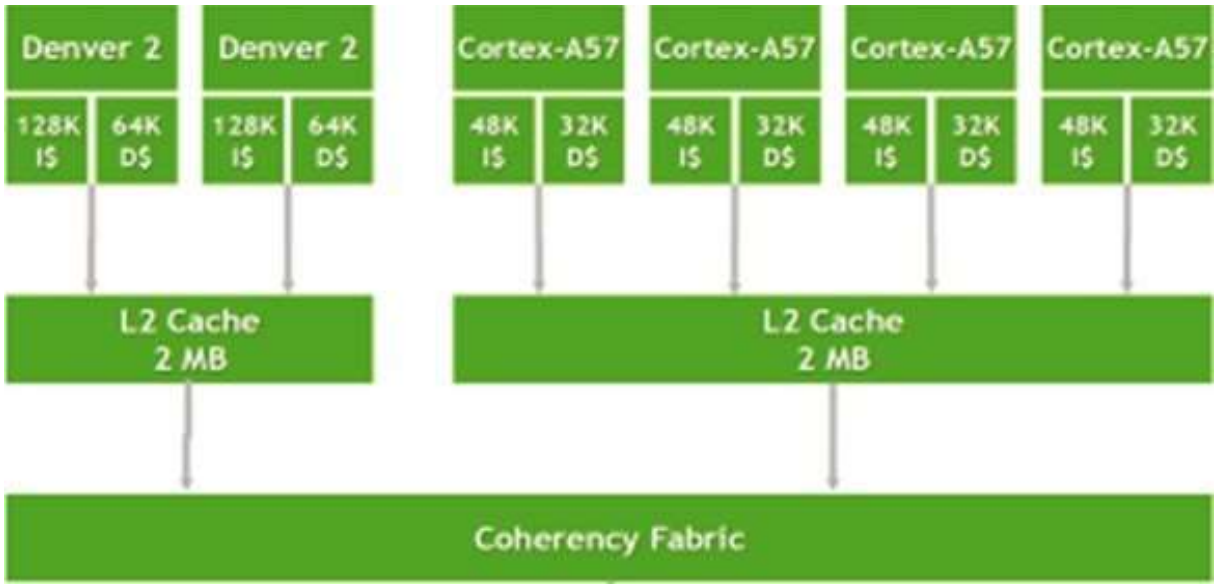
- The newest implementation of 4-plus-1 technology
- Introduced in 2014 in 28nm
- Uses Cortex A15 cores
- A 192-core Kepler GPU
- Many interfaces and display engines





nVidia Tegra X2

- Uses a new model called BigSuper



- Used in Nvidia Drive PX driver assistance functionality powered by deep learning

Outline

- 3D integration
- Processing-in-memory
- 2.5 D chips and interposer

Memory bottleneck

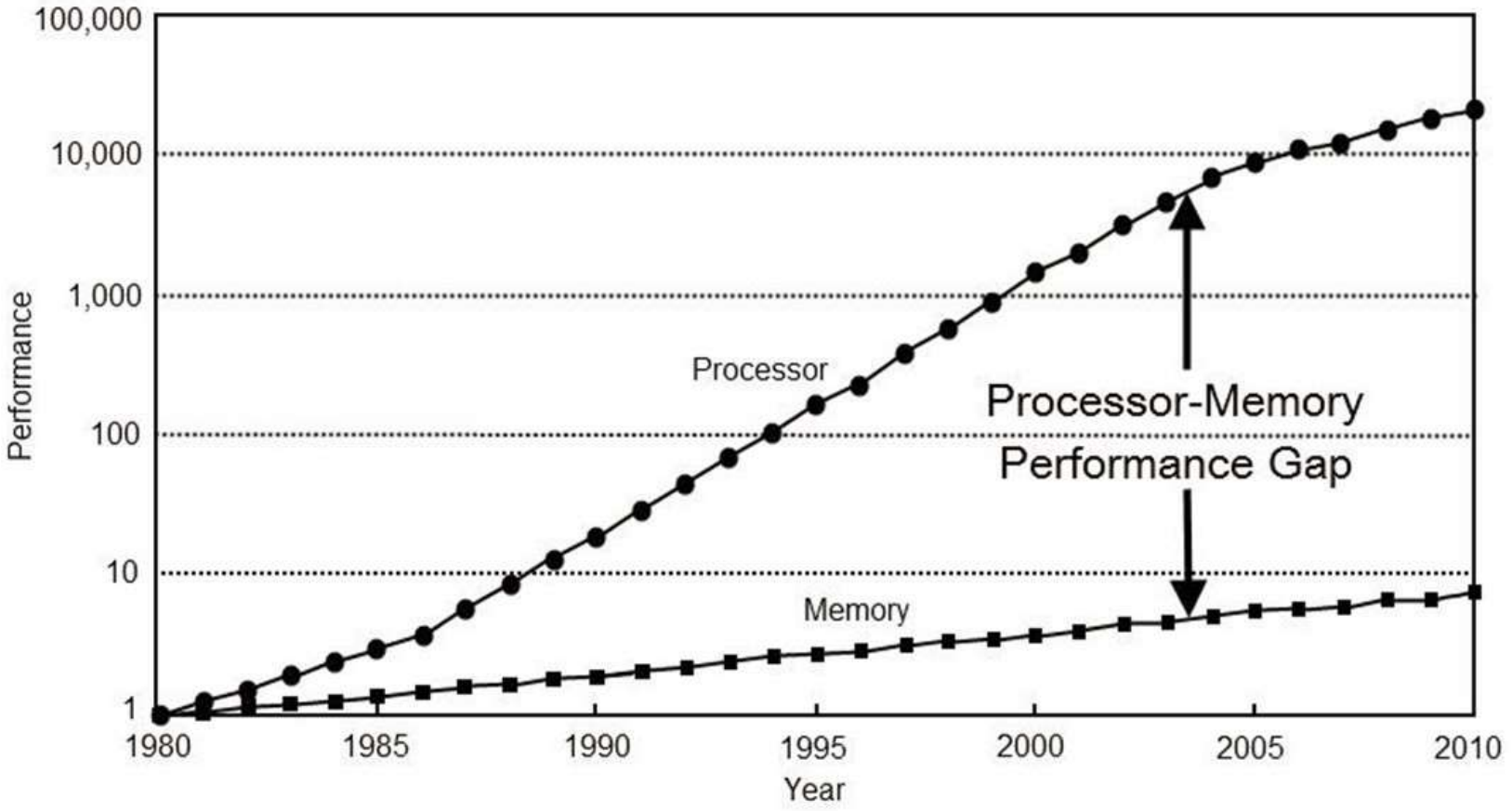
1.
   - Uses slow and power hungry on-board chips
   - Bound by the chip pinout (number of pins)
2. The memory speed
   - The speed of memory does not scale proportional to theCPU’s

Memory bottleneck- connections

- The connection between CPU and memory controller is no longer a bottleneck in modern systems
- But, the connection between memory controller and memory chips is still a bottleneck
  - Uses on-board links, limited by chip pinout
- Energy consumption due to data transfer overhead is not scaling down!

Memory bottleneck-memory

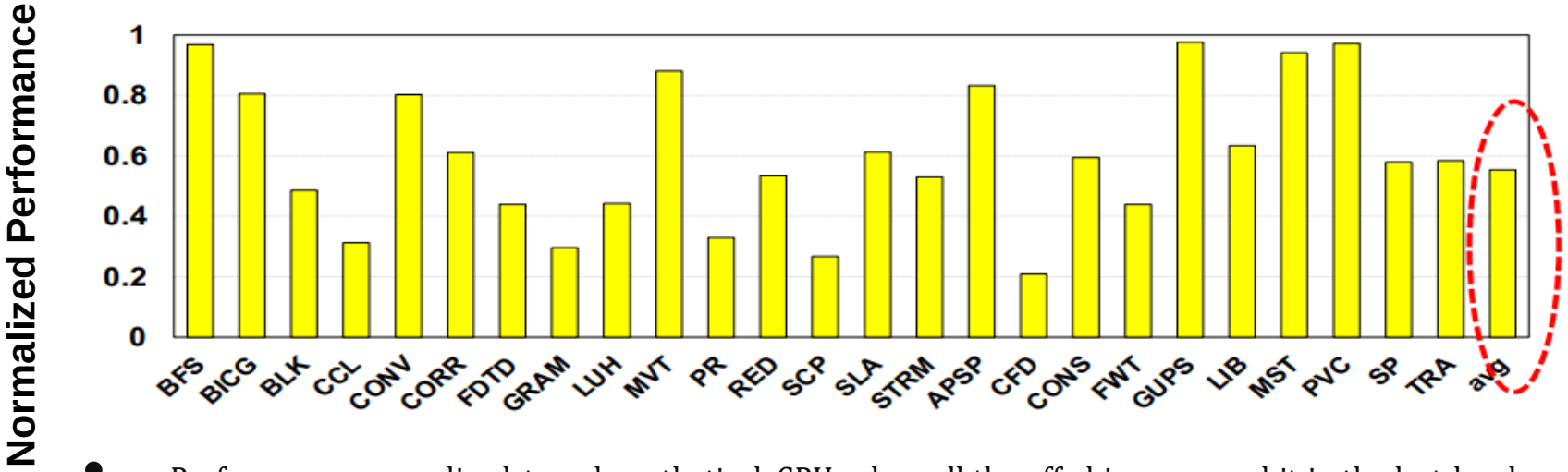
- Processor-memory performance gap



Memory bottleneck

- L3 and DRAM read latency on a 3.4GHz CPU
- DRAM is by far slower than cache. Why?
  - Logic VLSI Process: optimized for better transistor **performance**
  - DRAM VLSI Process: optimized for **low cost** and **lowleakage,** so slower

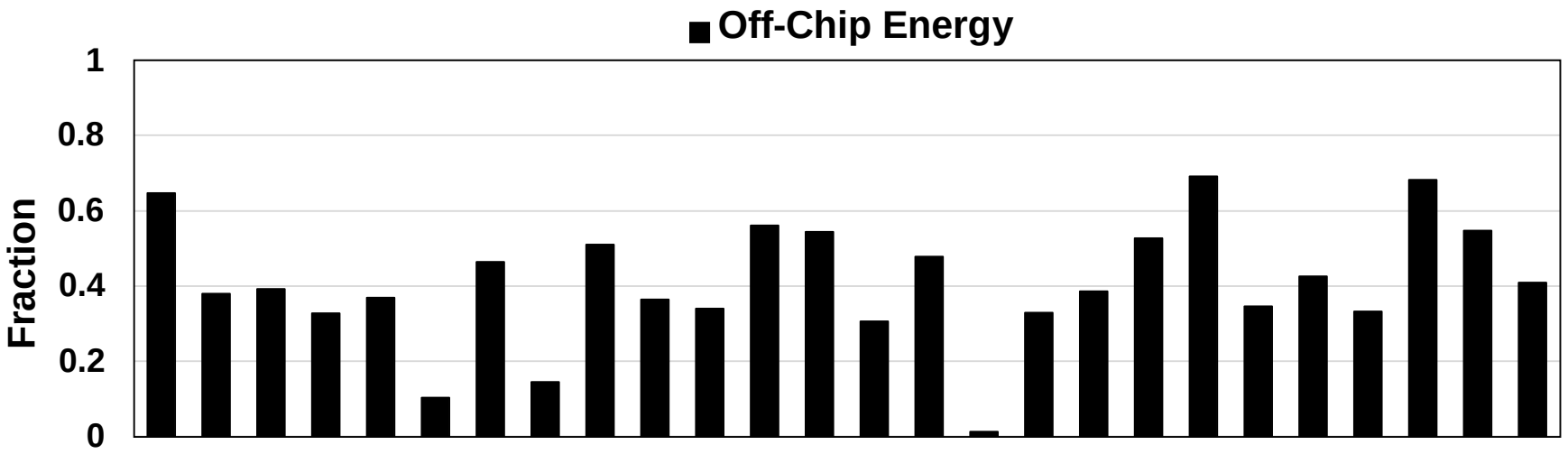
Memory bottleneck- performance



- Performance normalized to a hypothetical GPU whereall the off-chip accesses hit in the last-level cache
- Main memory accesses lead to 45% performance degradation.

Source: Onur Mutlu, PACT'16

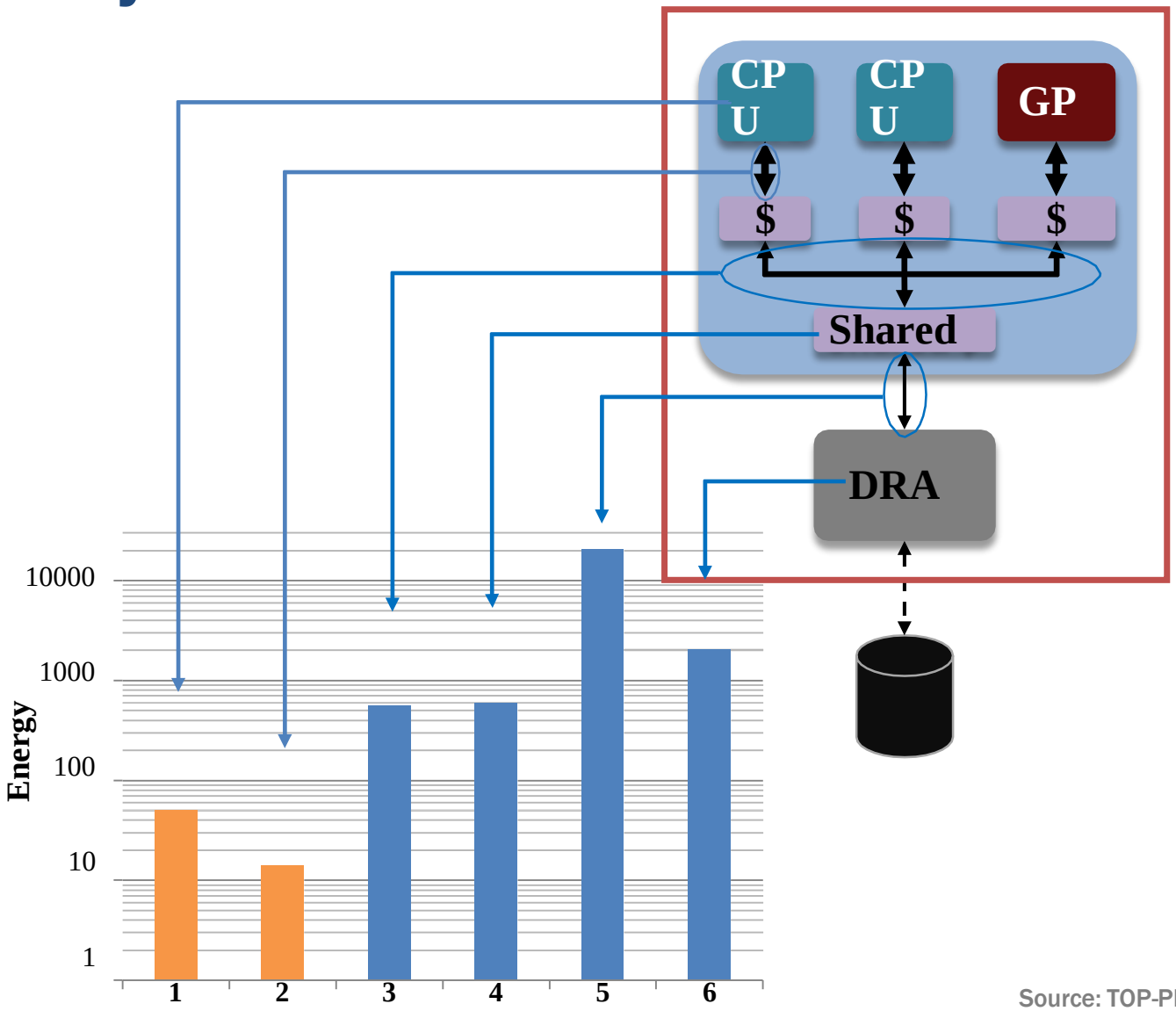
Memory bottleneck- power



Source: Onur Mutlu, PACT'16

- Data movement energy consumption caused by off-chipmemory accesses.
- 41% of total energy consumption of the system (on average)

Memory bottleneck- Power



Source: TOP-PIM, AMD research

Memory bottleneck- Solution

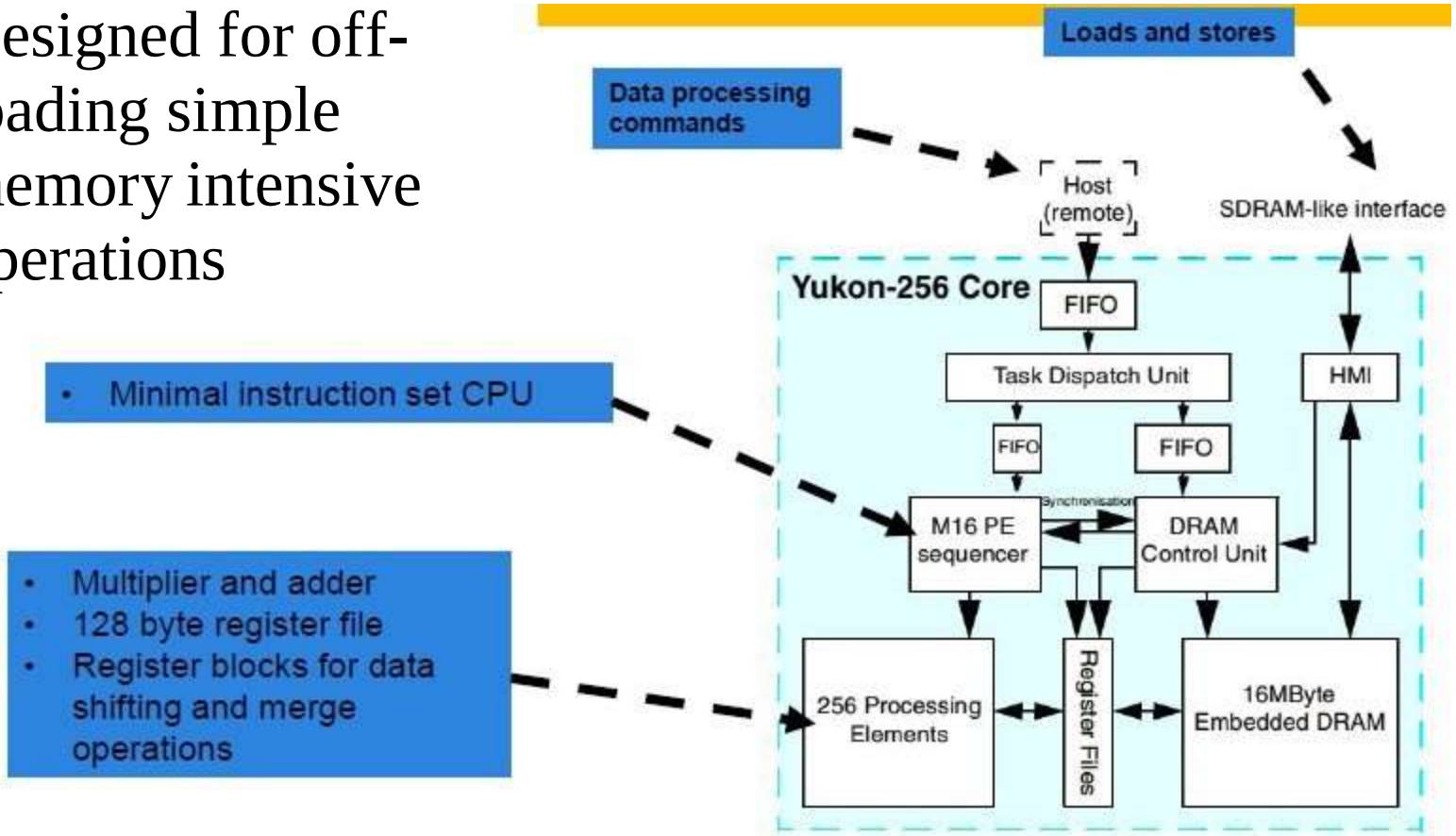
- Store data closer to the processor
- Processing in memory (PIM) or Near Data Processingconcepts
  - Implement processors on memory
  - Integrate memory into the processor chip

Early Processing-in-Memory systems

- Placing processing units on same die with DRAMprovides increased bandwidth
- Merged Logic and DRAM (MLD) process was emergingas a result od multiple efforts from industry and academia
  - IBM, Mitsubishi, Samsung, Toshiba and others
  - Example: Micron's Active Memory(Yukon), UC Berkeley'sIRAM, Stanford's Smart Memories, UIUC'FlexRAM, and many more….

An old PIM: Micron Yukon (2002)

- 0.15µm DRAM/0.18µm embedded Logic
- Implement logic on a dense DRAM die
- Designed for off-loading simple memory intensive operations



Why early PIM efforts failed?

- Merged Logic and DRAM (MLD) process did not grow
- Early PIM proposals:
  - **Logic embedded on modified DRAM process**: substantially slower logic, typically multiple processgenerations behind contemporary logic processes
  - **DRAM embedded on modified logic process**: leakytransistors, high refresh rates, increased cost/bit (increased manufacturing complexity)

DRAM scaling problem

- DRAM stores charge in a capacitor (charge-based memory)
- Capacitor must be large enough for
  - reliable sensing
  - Tolerance against noise and SEU
- Access transistor should be large enough for low leakage and high retention time
- " Scaling beyond 40-35nm (2013) is challenging [ITRS,2009]

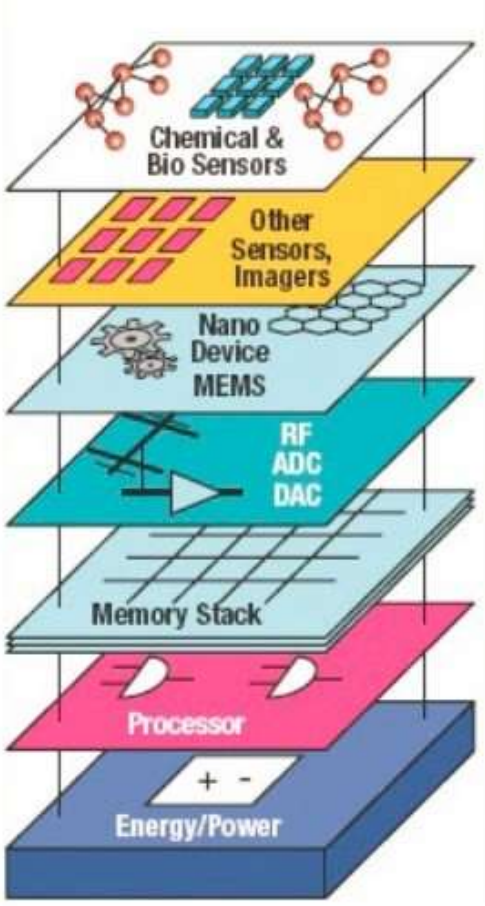
Memory bottleneck- Solution

- It's a promising approach to minimize data movement.
- The concept dates back to the late 1960s
- Technological limitations of integrating fast computational units in memory was a challenge

- Significant advances in adoption of 3D-stacked memory has enabled tight integration of memory dies and logic layer
  - Brought computational units into the memory stack

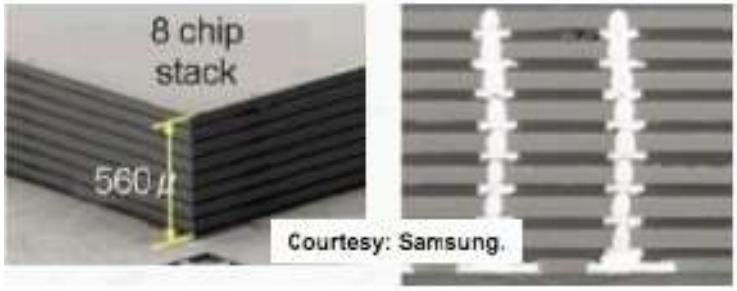
3D integration

- Different devices are stacked on top of each other
  - Each layer can use different process technology
- Layers are connected by through-silicon vias (TSVs)
- TSV: a vertical electrical connection passingcompletely through a silicon wafer or die
  - Shorter conductor
  - Less capacitance
  - Potentially increased signalling rate over longer metallic interconnects in 2D



3D integration challenges

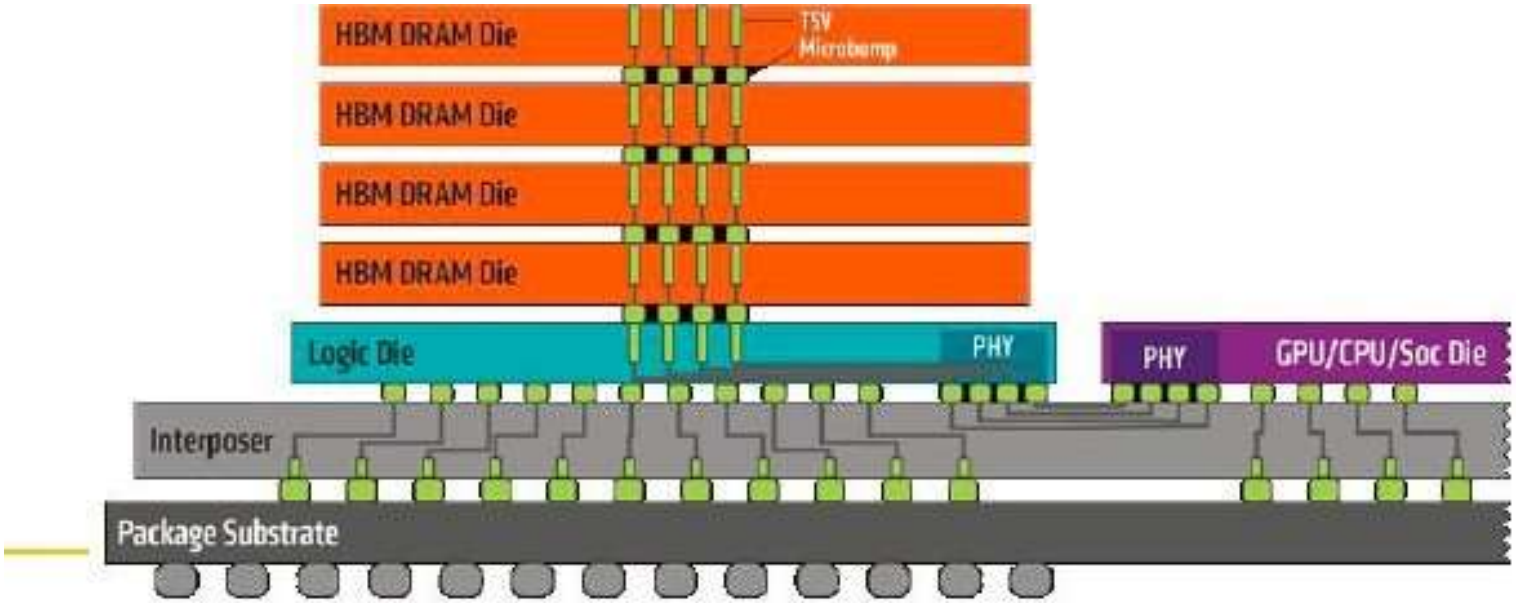
- Removing heat from inner layers is challenging
- DRAM requires doubling the refresh rate for temperatures above 85C
- Supplying power to all layers



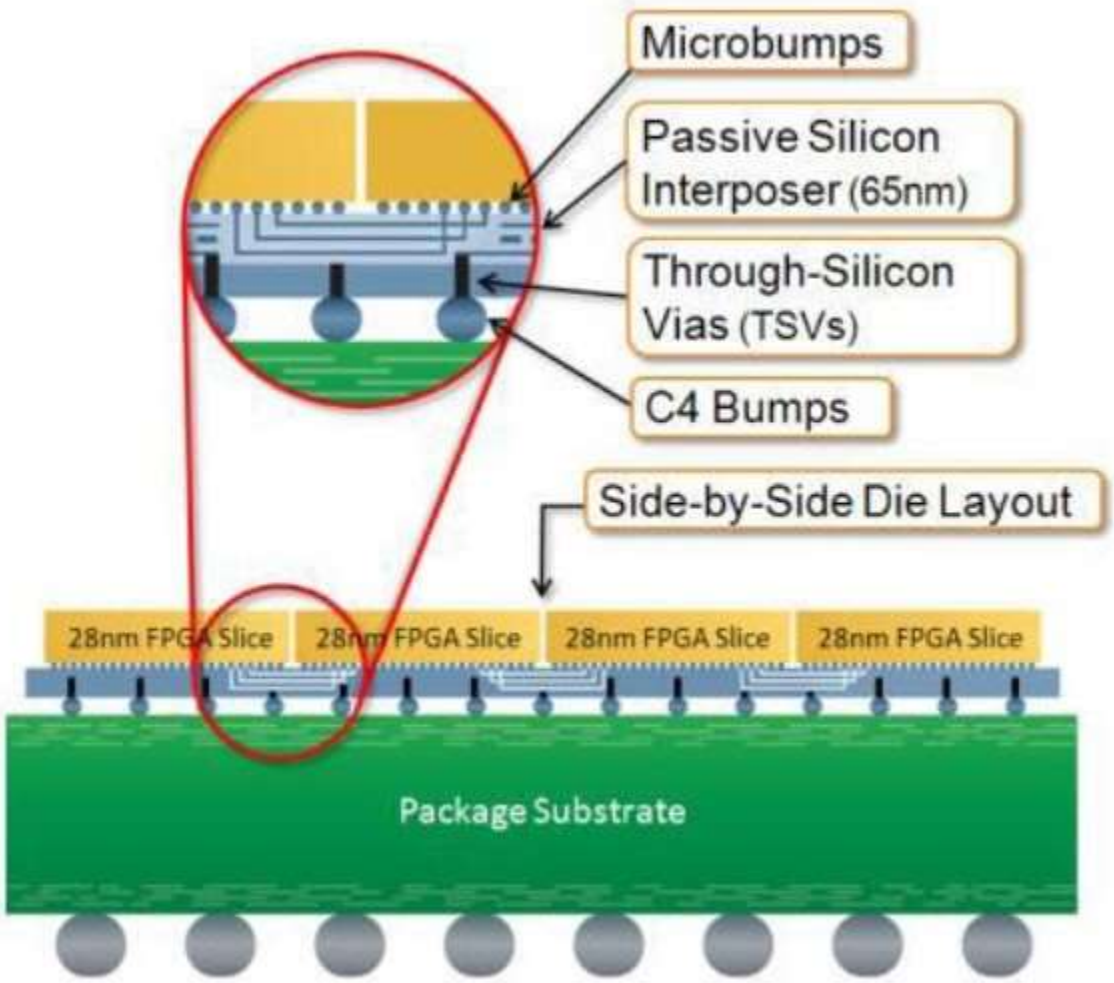
3D DRAM Memory

2.5D integration

- Building a 3D chip has been challenging
- Industry came up with an “evolutionary” design
- Different chips are placed on a passive silicon layer (the interposer)



2.5D chip components



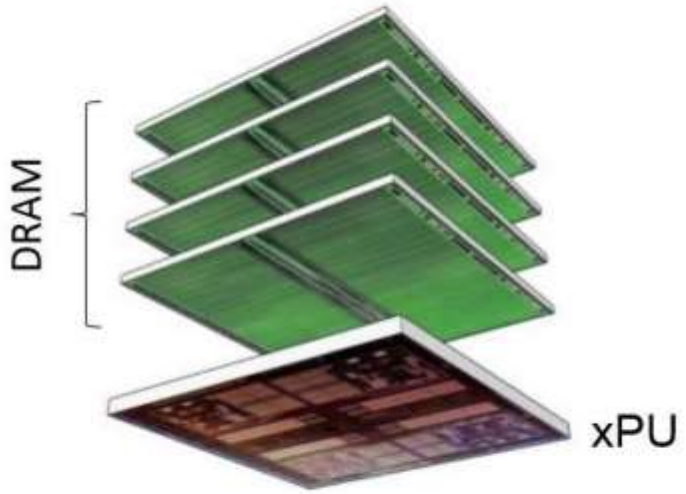
2.5D chip components

- With 2.5D stacking, chips are typically mounted face down on the interposer with an array of micro-bumps (μbumps)
  - μbumps provide electrical connectivity from the stacked chips to the metal routing layers of the interposer
- Chips stacked horizontally can communicate with each other with electrical connections on an interposer:
  - Start from a source chip's top-level metal
  - Through a μbump
  - Across a metal layer on the interposer
  - Back through another μbump
  - Finally to the destination chip's top-level metal

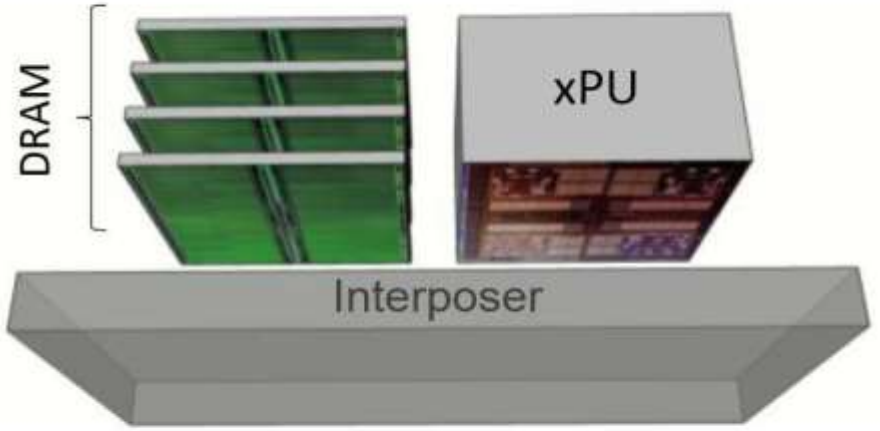
3D vs 2.5D memory

BUT FIRST… THERE ARE TWO FLAVORS OF DIE STACKING

AMD



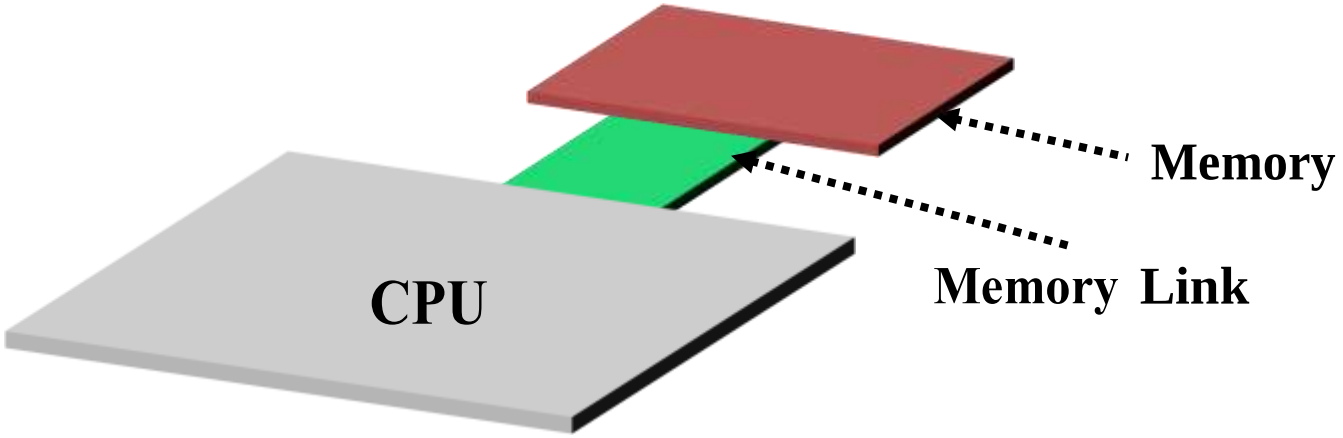
VERTICAL STACKING (3D)



INTERPOSER STACKING (2.5D)

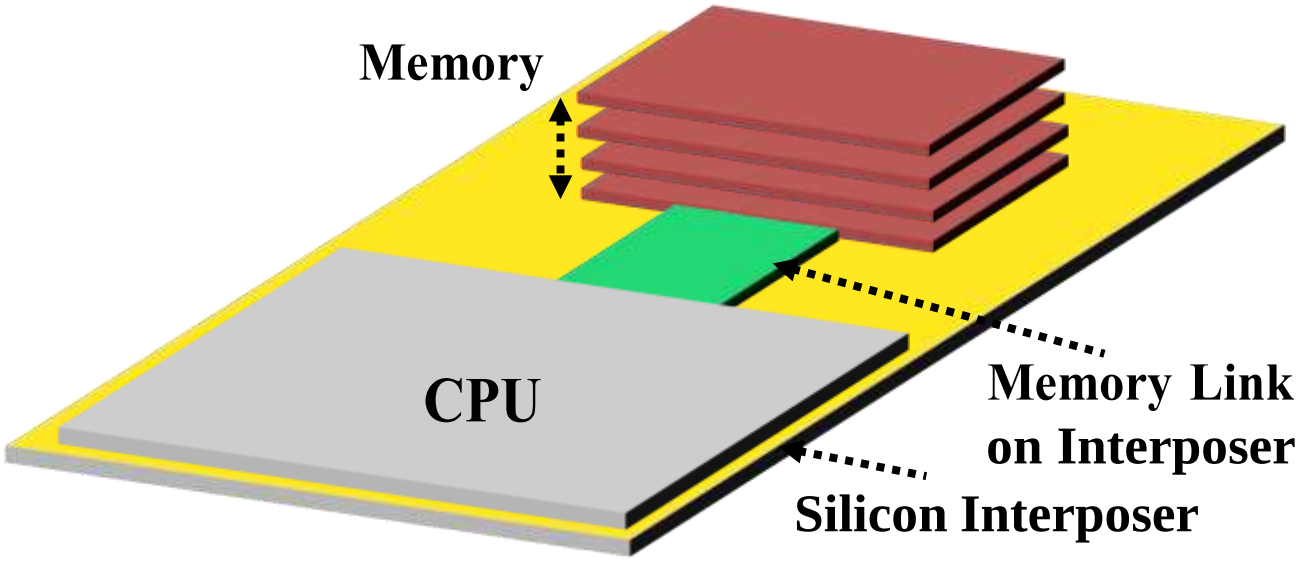
PIM-Assisted GPU architecture

- Traditional memory-CPU Connection



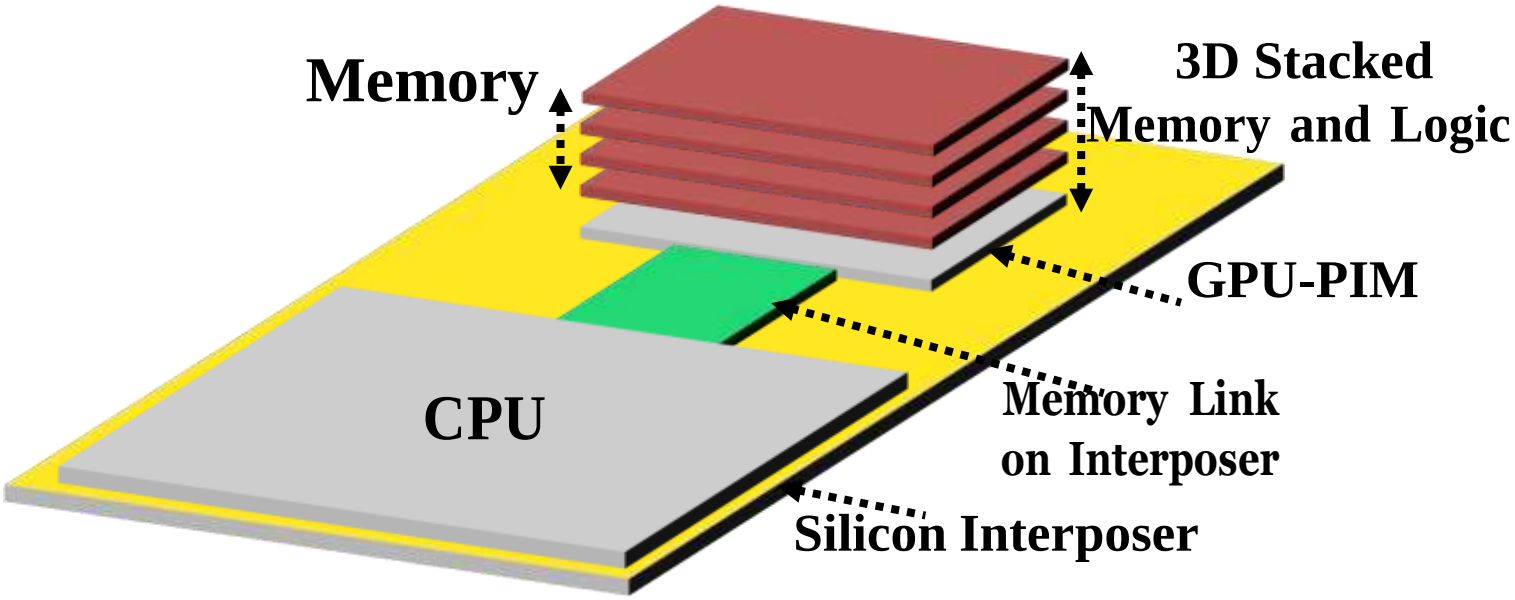
PIM-Assisted GPU architecture

- GPU architecture with 3D-stacked memory on a silicon interposer



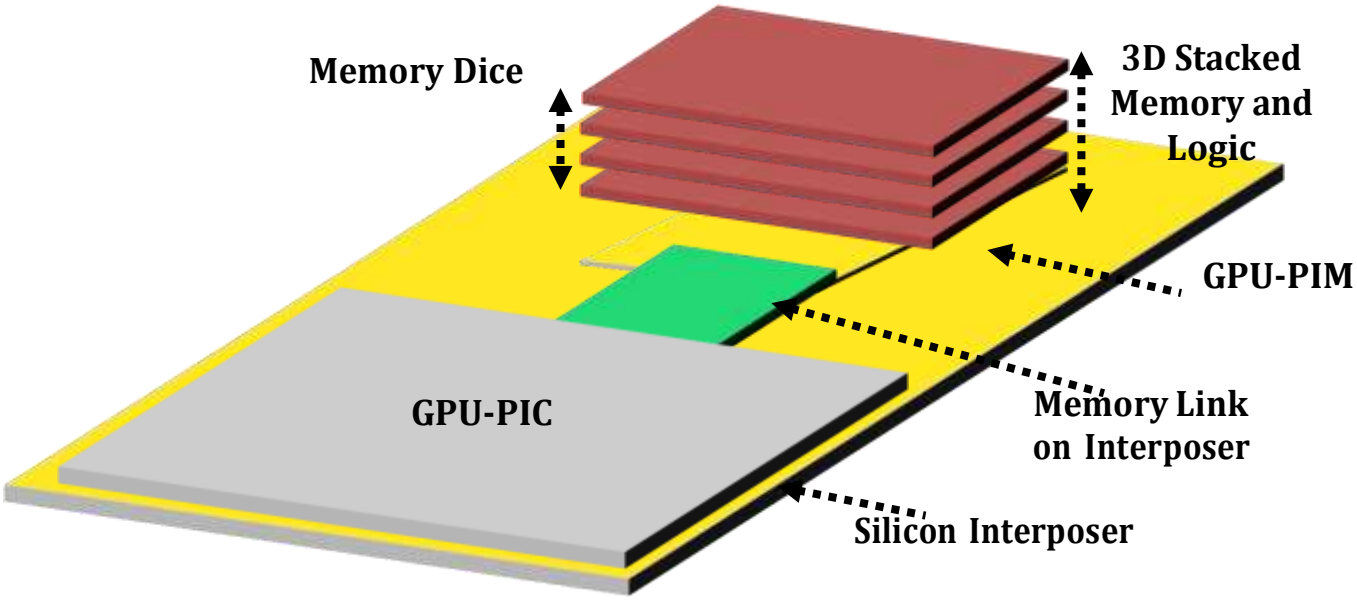
PIM-Assisted GPU architecture

- Now we add a logic layer to the 3D-stacked memory and we call this logic layer as CPU-PIM.



PIM-Assisted GPU architecture

- Application can now be run on both GPU-PIC and GPU-PIM
- Challenge: Where to execute the application on?



2.5D integration

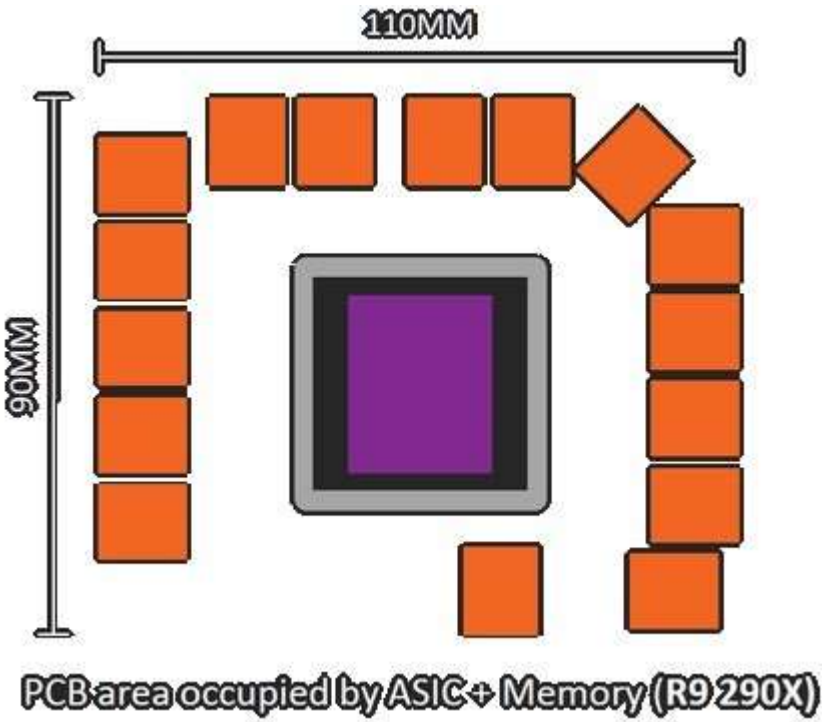
- Why is this better than 3D?
  - The bottom layer(the interposer) is not active
  - Less heat dissipation
- Why is this better than a PCB?
  - Interconnects can be placed closely
    - Increased I/O density

High Bandwidth Memory (HBM)

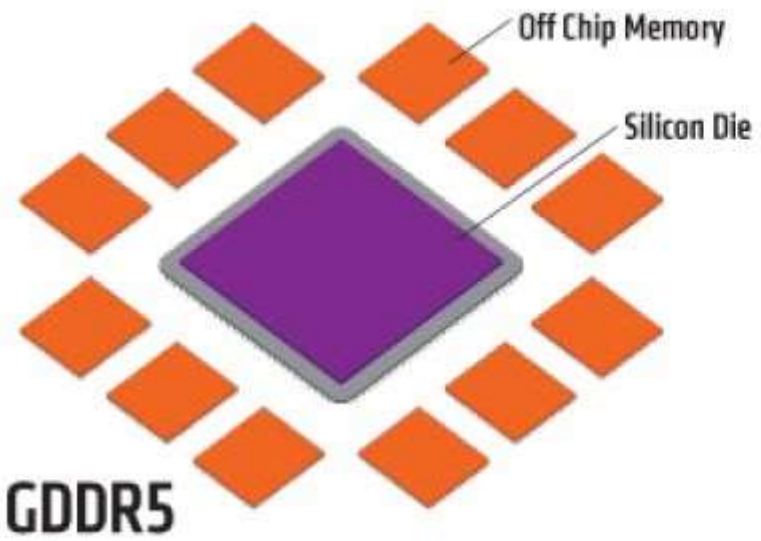
- The first and most successful industrial 2.5 D memory implementation
- Initiated by AMD
- Examples:
  - AMD Fiji GPU
  - Intel Xeon Knights Landing
  - Virtex 7 FPGAs

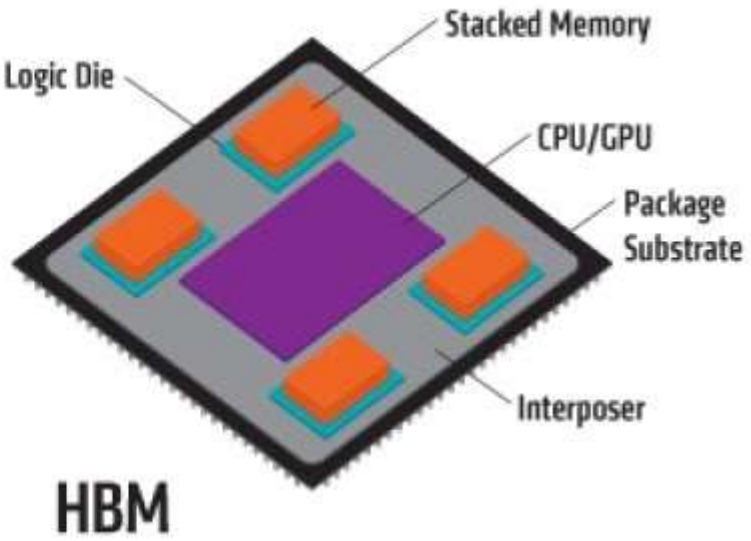
DDR vs HBM

- AMD Radeon R9 desktopGPU, produced in 2013
- Used GDDR5 ram
- GDDR: high bandwidthmemory for GPUs



DDR vs HBM



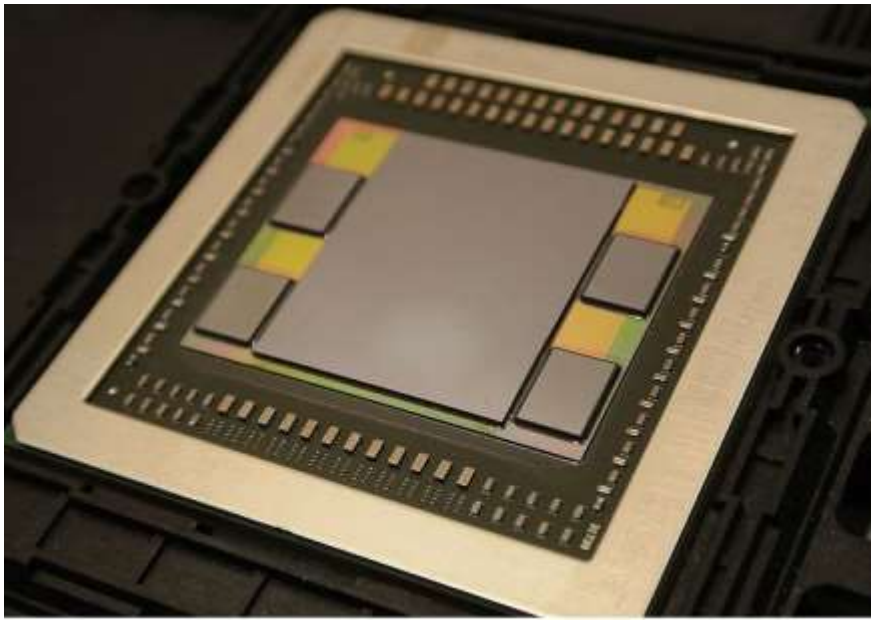


Bus Width: 32 bit
Clock Speed: 1750MHz
Transfer Rate per pin: 7GB/s
Bandwidth: 28GB/s per chip
Bandwidth per Watt: 10.5GB/W

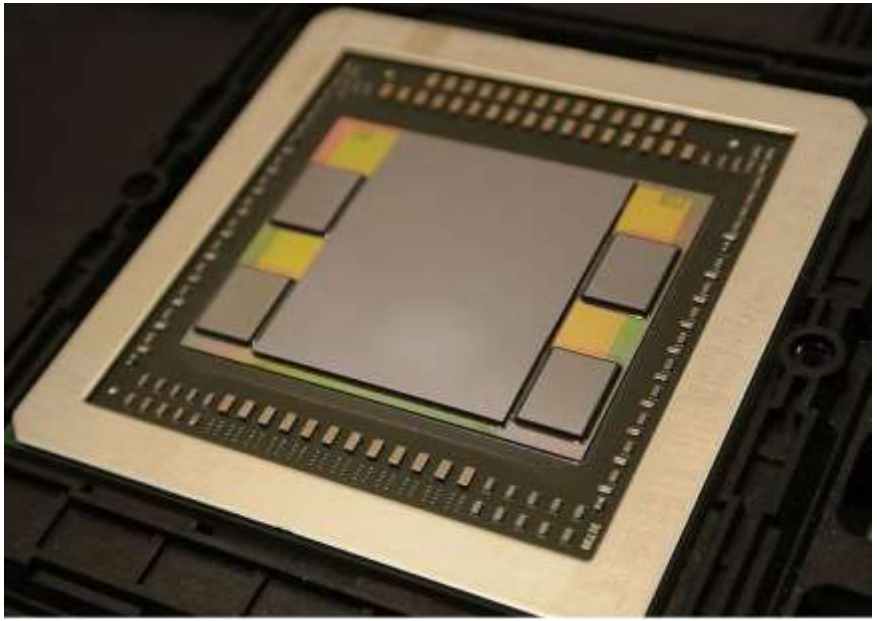
- Bus Width: 1024 bit
- Clock Speed: 500MHz
- Transfer Rate per pin: 1GB/s
- Bandwidth: 128GB/s per chip
- Bandwidth per Watt: 35GB/W

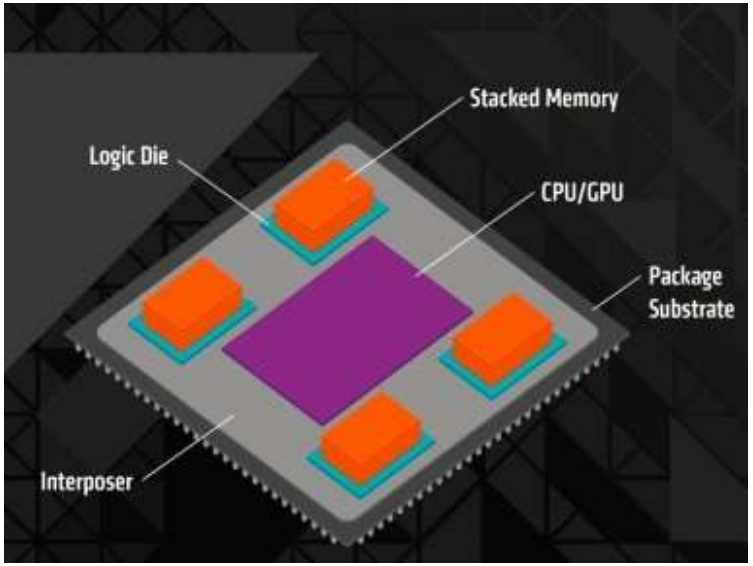
AMD Fiji

- AMD Radeon R9 Fury (codename: Fiji)
- GPU and memory are connected by an interposer layer

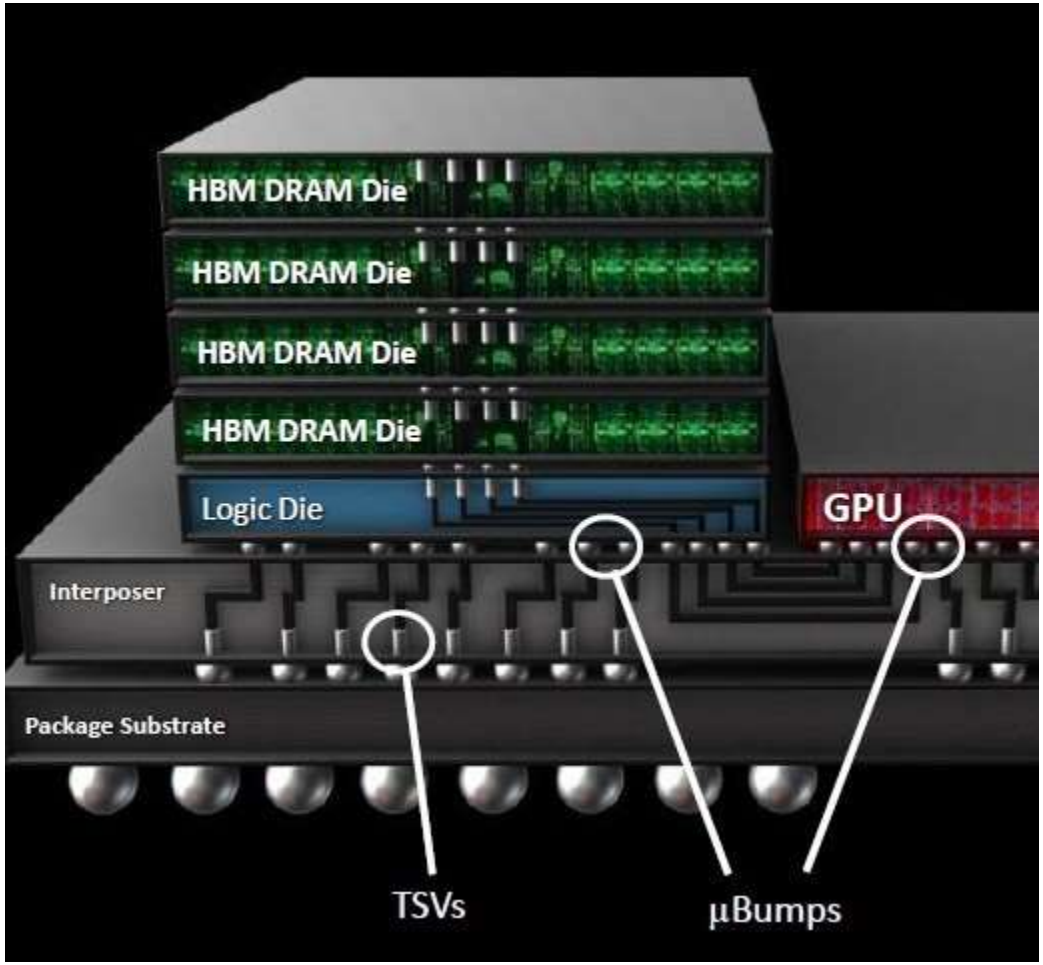


AMD Fiji- under the package



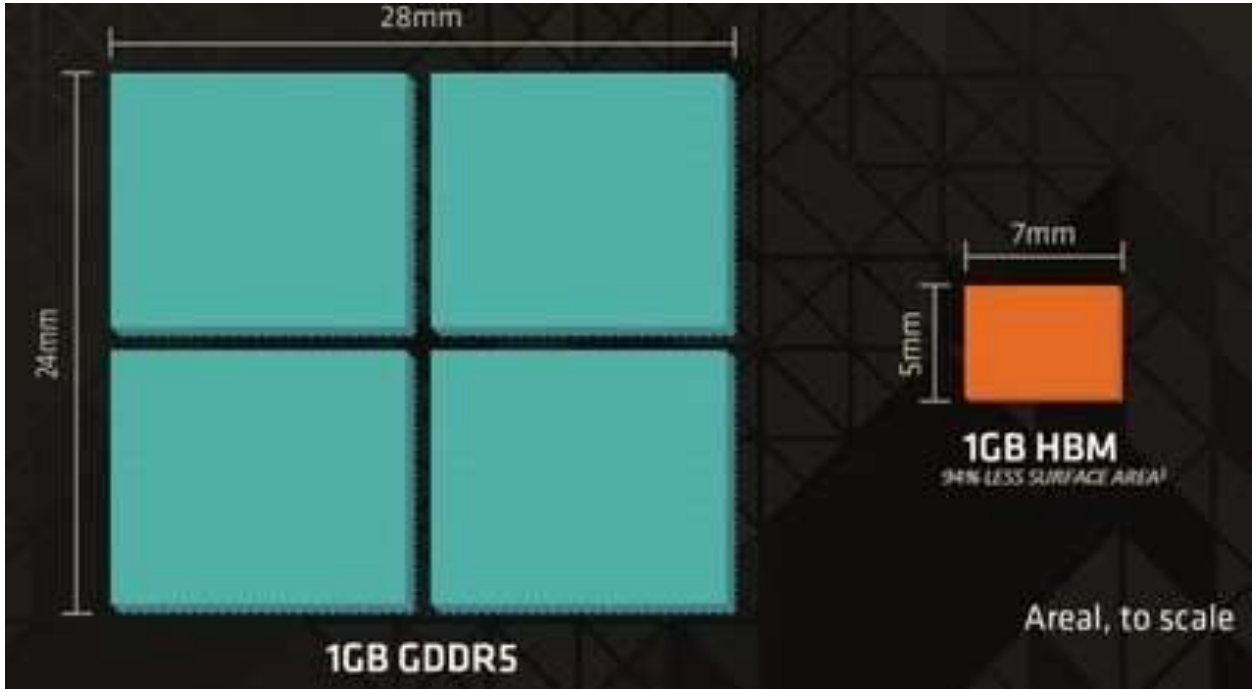


AMD Fiji- under the package



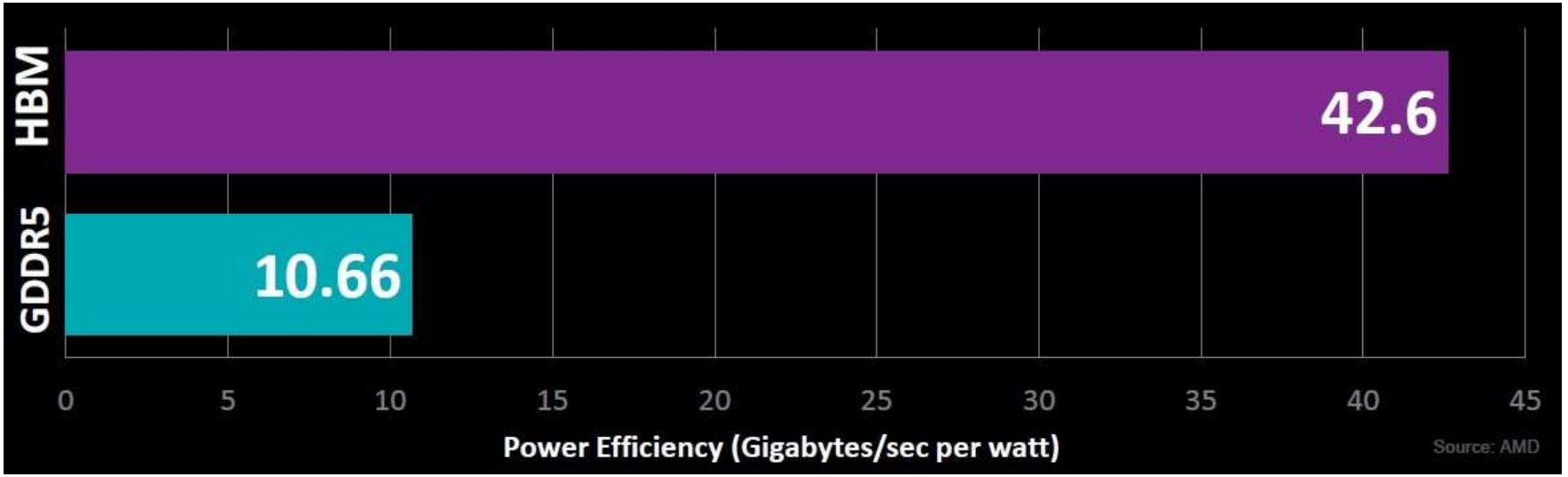
DDR vs HBM

- 1 Gbyte DDR5 vs 1GByte 3D memory
- 3D is faster with smaller area



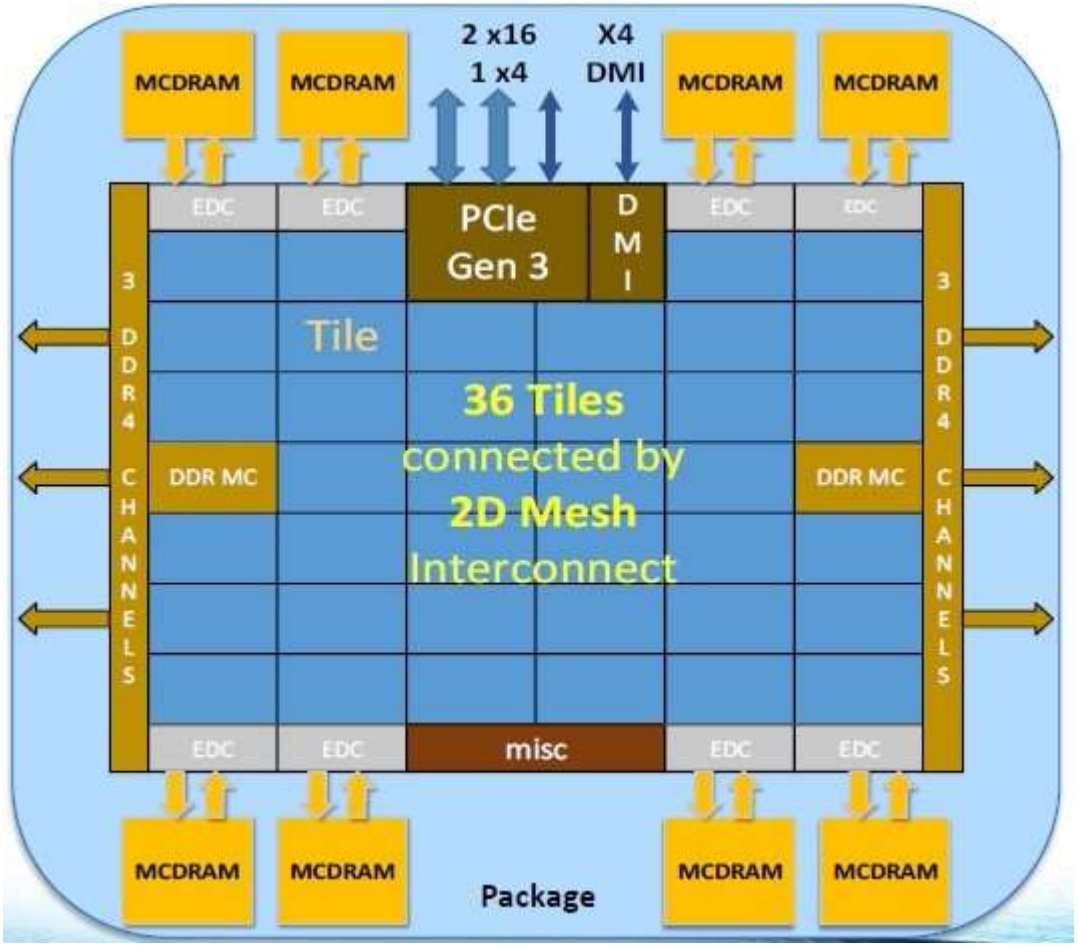
DDR vs HBM

- HMB is more power efficient



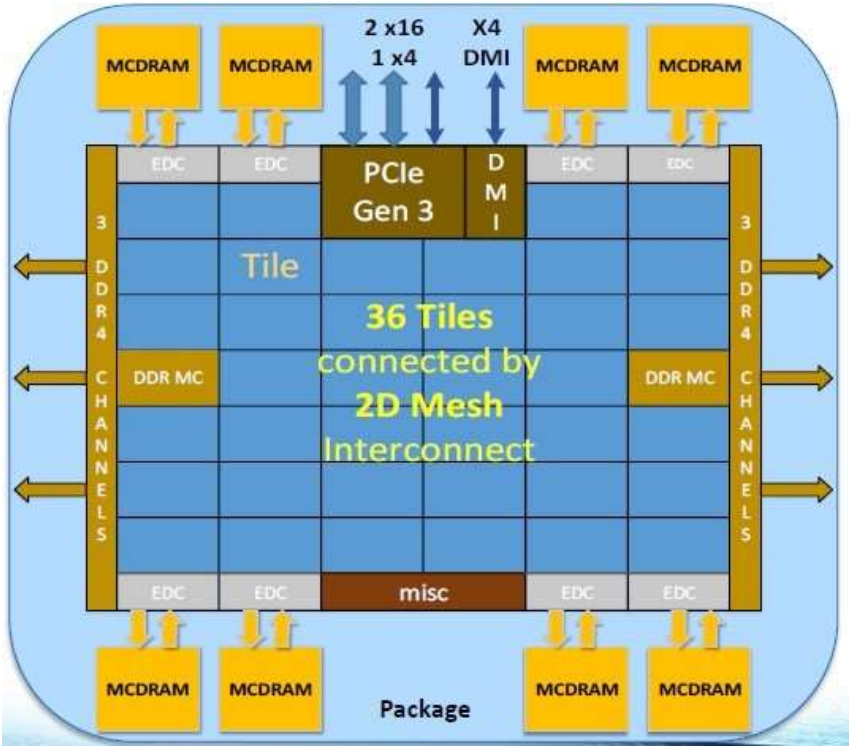
Intel Xeon Knights Landing

- CMP with 36 tiles and 8 on-package memory slides



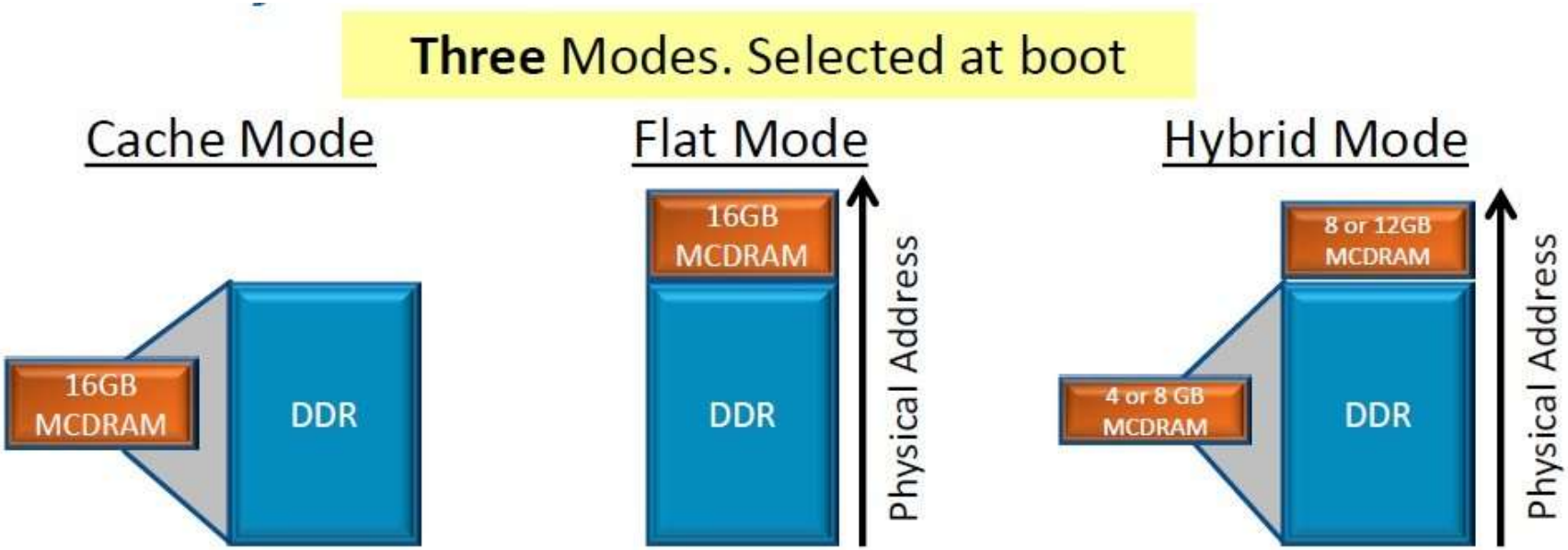
Intel Xeon Knights Landing

- MCDRAM: Multi-ChannelDRAM
  - On-chip RAM
- Main memory is still off-chip and is accessed by 6 DDR4 channels



Intel Xeon Knights Landing

- MCDRAM: 8-16 GB on-chip RAM
- Can be used as cache or part of memory

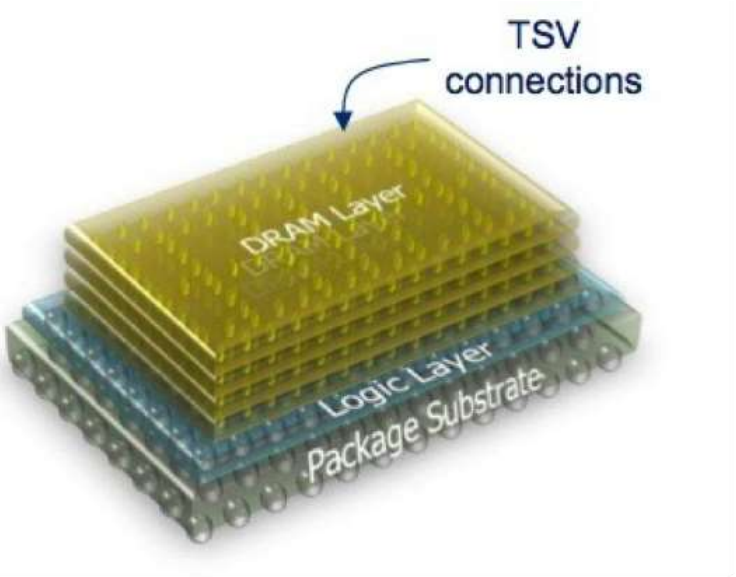


hybrid memory cube

- Hybrid Memory Cube wasannounced by Micron Technology in 2011
- 15 times speed improvement over DDR3
- Now is backed by several major technology companies including Samsung, Micron Technology, Open-Silicon, ARM, and Xilinx

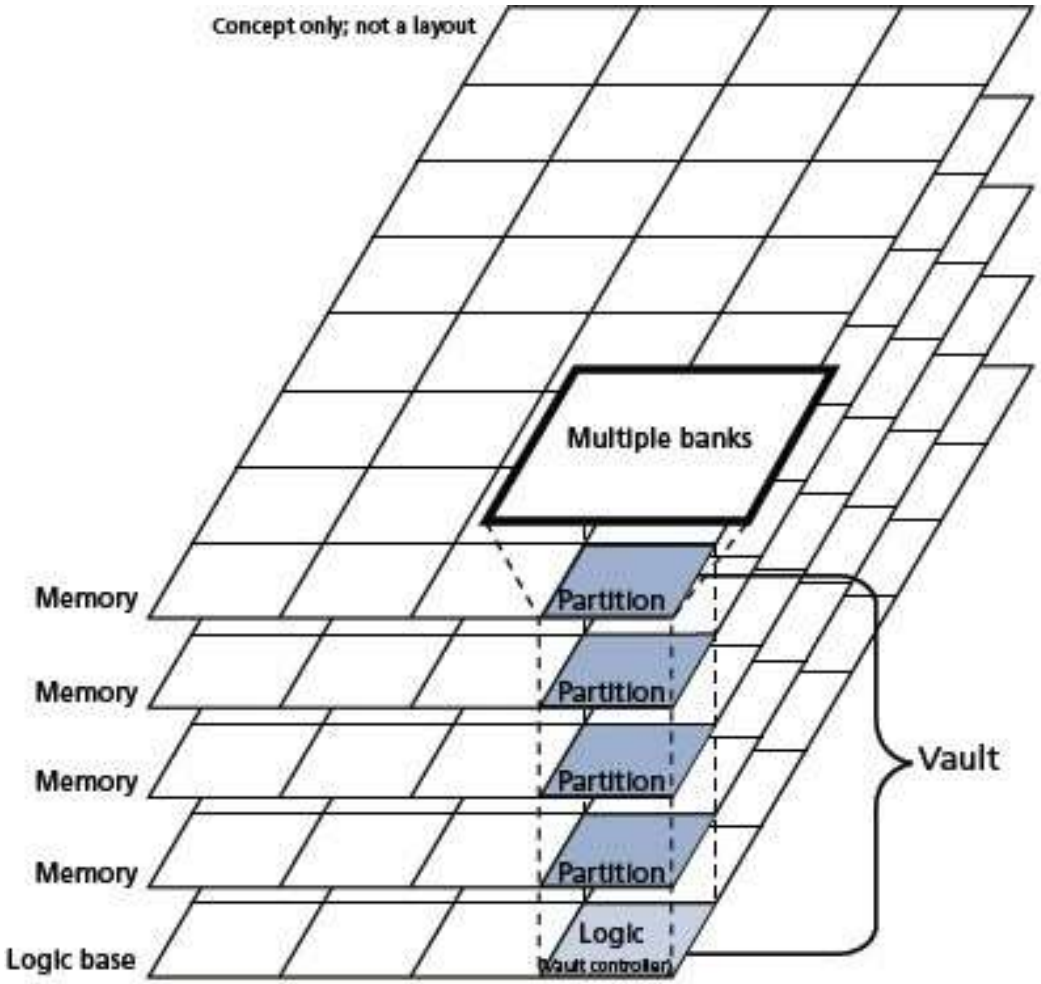
HMC

- An HMC consists of a single package containingmultiple memory die and one logic die
- stacked together using through-silicon via (TSV)



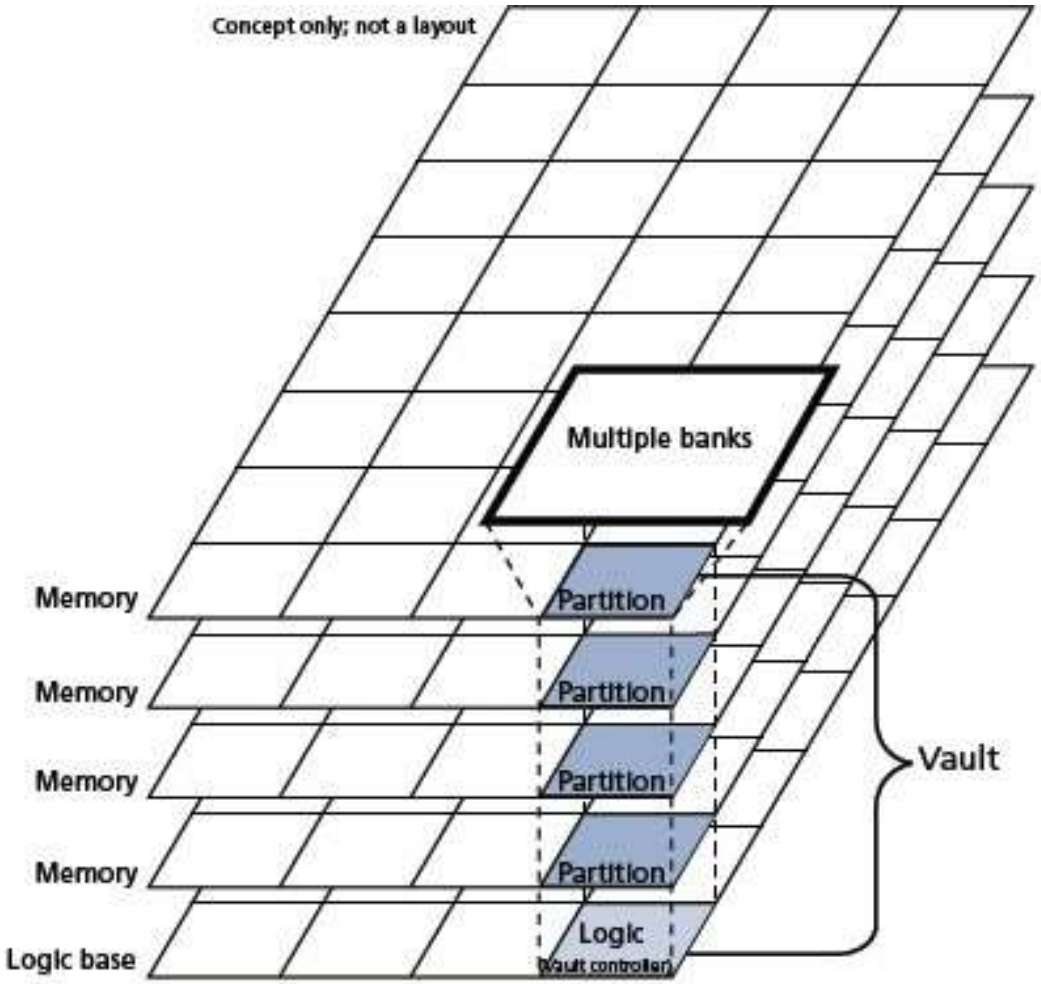
HMC structure

- Within an HMC,memory is organized into vaults
- Each vault is functionally and operationally independent



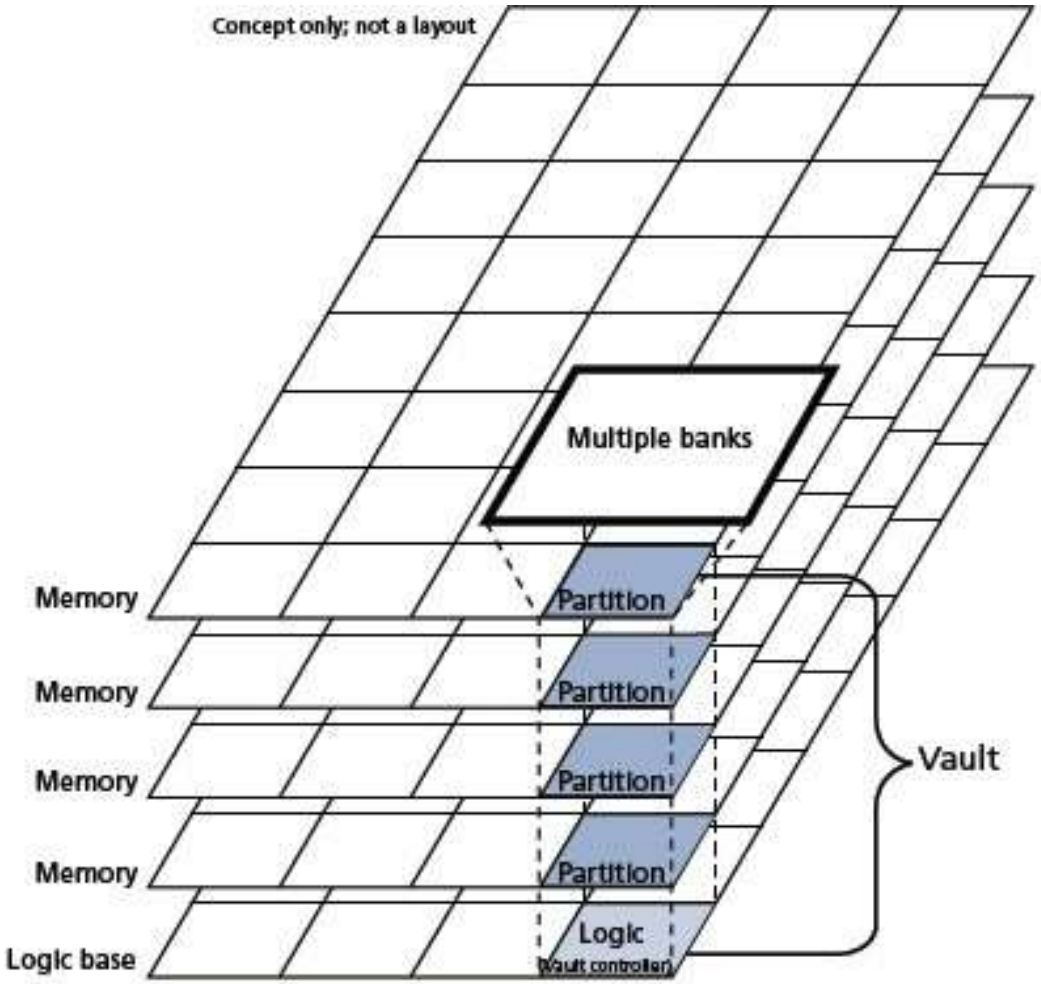
HMC structure

- Each partition of a vault is a bank
- Banks can be accessed (almost) in parallel
- Figure shows 324-slice vaults
- Each slice has twobanks



HMC structure

- Each vault has a memory controller (called a vault controller) in the logic base that manages memory reference operations within that vault
- Refresh operations are done by the vault controller

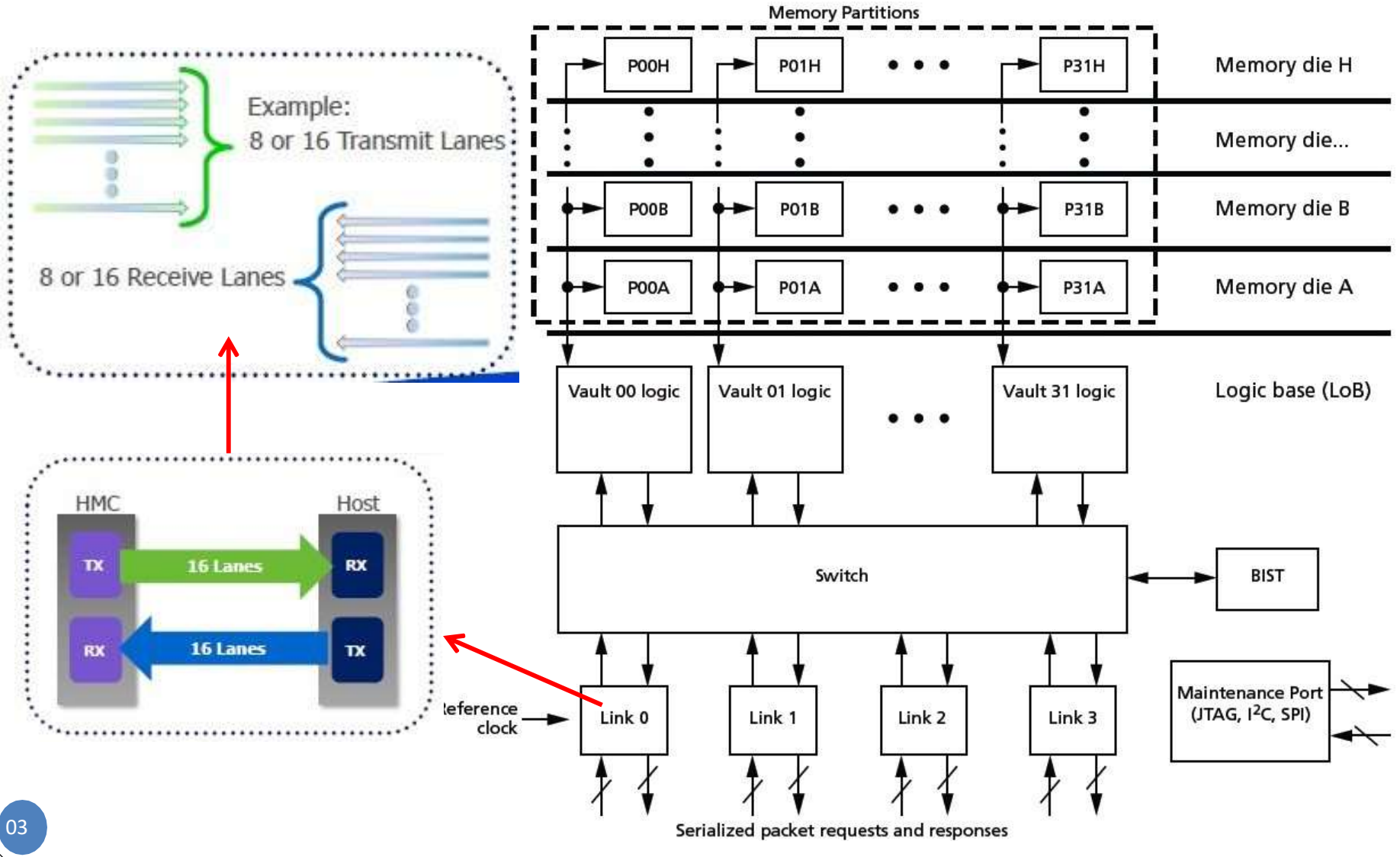


Memory address

- Memory address should specify vault, bank, block, and byte within block

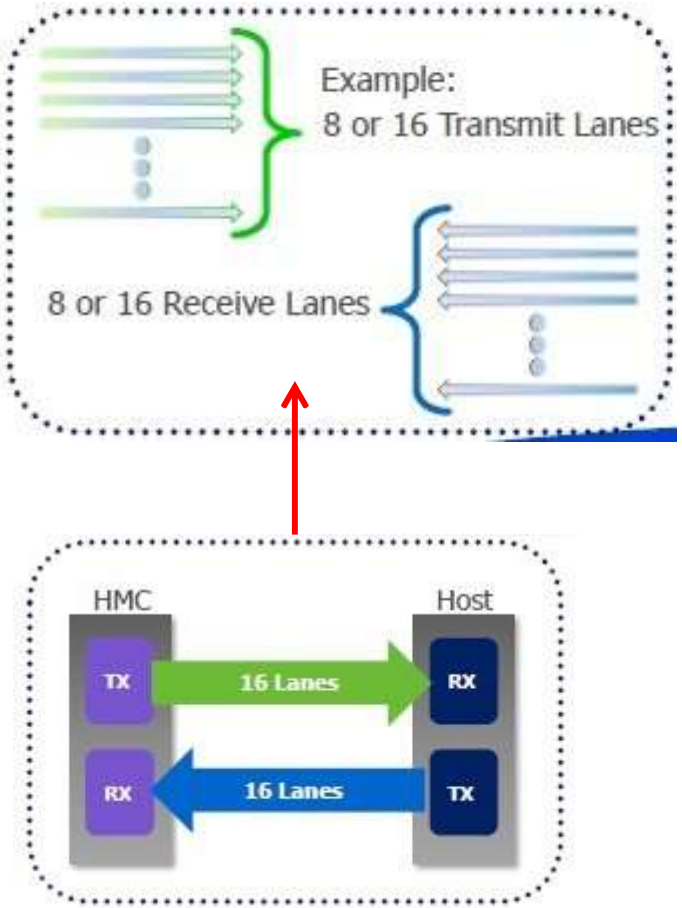
Address	Description	
Byte address	Bytes within the maximum supported block size	4 bit to specify 16 bytes in eachblock. Used only for operations.
Vault address	Addresses vaults within the HMC	5 bits to specify 32 vaults
Bank address	Addresses banks within a vault	3-bit for 8-bank (in 4GB HMC)and 4 bits for 16-bank (in 8GBHMCs)
DRAM address	Addresses DRAM rows and column within a bank	20 bits for 1MB blocks in eachbank

HMC block diagram



HMC block diagram

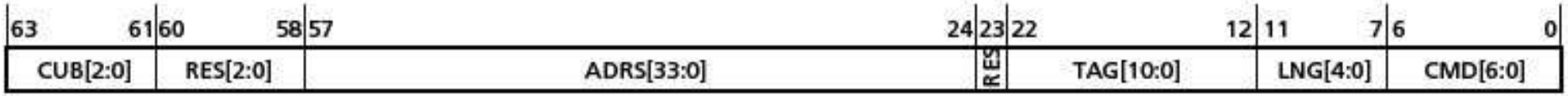
- Each memory module has up to 240 GB/s bandwidth
  - Each link has up to 60 GB/sbandwidth
  - Each cube (memory module) has four links



HMC to processor communication

- HMC and processor use packets to send/receive data
- Each packet has a header and several data parts
- Request packets: carry read/write/in-memory operations from processor to memory
- Response: the response of the request packet that goes from HMC to processor

Header of a request packet



Name	Field Label	Bit Count	Bit Range	Function
Cube ID	CUB	3	[63:61]	CUB field used to match request with target cube. The internal Cube ID Register defaults to the value read on external CUB pins of each HMC device.
Reserved	RES	3	[60:58]	Reserved: These bits are reserved for future address or Cube ID expansion. The responder will ignore bits in this field from the requester except for including them in the CRC calculation. The HMC can use portions of this field range internally.
Address	ADRS	34	[57:24]	Request address. For some commands, control fields are included within this range.
Reserved	RES	1	[23]	
Tag	TAG	11	[22:12]	Tag number uniquely identifying this request.
Packet length	LNG	5	[11:7]	Length of packet in FLITs (1 FLIT is 128 bits). Includes header, any data payload, and tail.
Command	CMD	7	[6:0]	Packet command. (See Table 23 (page 51) for the list of request commands.)

HMC instructions

- In addition to read and write, some simple in-memory operations can be requested in the CMD field of a request packet

Type	Data Size	Operation
Arithmetic	Dual 8B	Add immediate
	Single 8B	Add immediate
	8-byte	Increment
Bitwise	16-byte	Swap
	8-byte	Bit write
Boolean	16-byte	AND
	16-byte	NAND
	16-byte	OR
	16-byte	XOR
Comparison	8-byte	CAS/if equal
	16-byte	CAS/if zero
	8/16-byte	CAS/if greater
	8/16-byte	CAS/if less

HMC instructions

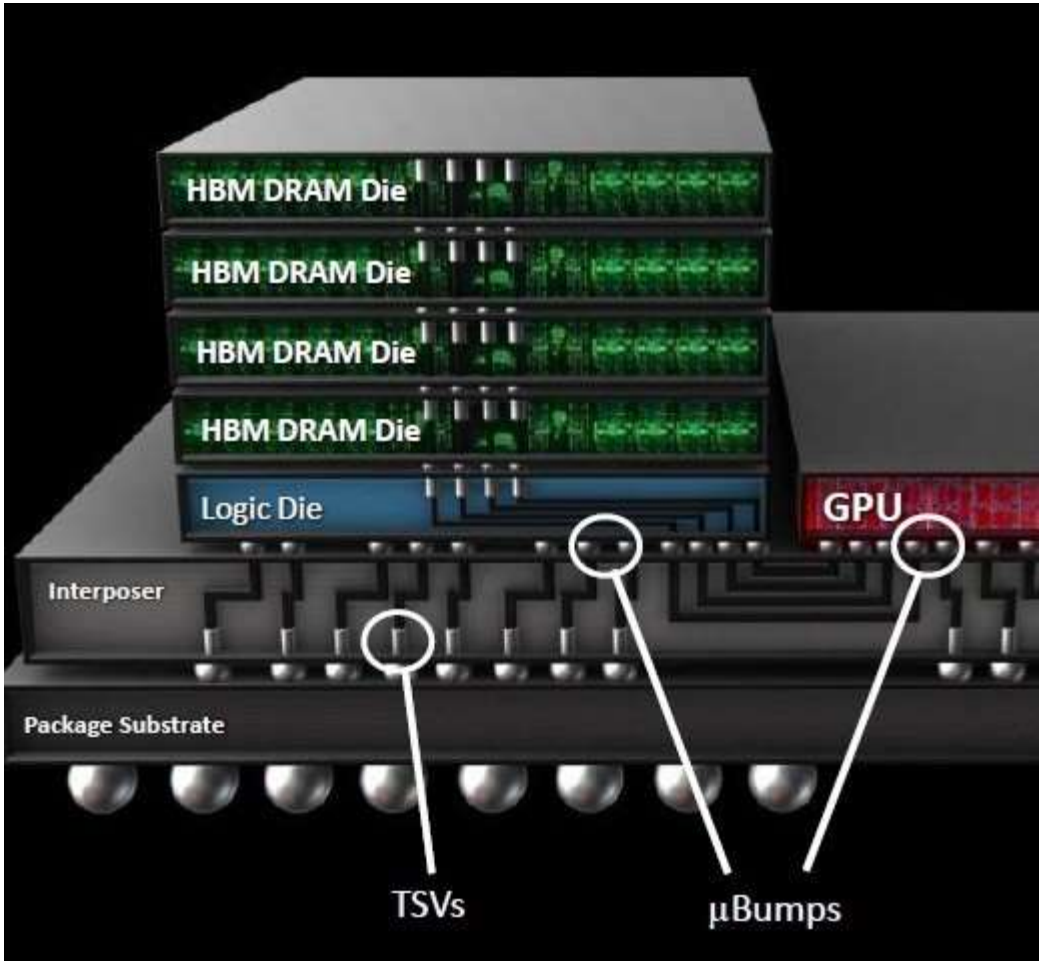
- Starting from HMC 2.0, atomic requests will besupported by standard HMC design
- The atomic requests include three steps, reading 16 bytes of data from DRAM, performing an operation on the data, and then writing back the result to the same DRAM location.
- All three steps occur in an atomic way
- Only one memory location operand is allowed
  - operations have to be performed between an immediate and a memory operand.

HMC instructions

- HMC Atomic Instruction
  - Instruction-level offloading
- Limitations
  - One-memory location
  - No floating-point
  - No indirect access
  - Simple operations only

HMC vs HMB

- HMB: High Bandwidth Memory
- Used in AMD Fiji
- HBM just used with interposer
- HMC is a separate chip connected to the CPU on the PCB

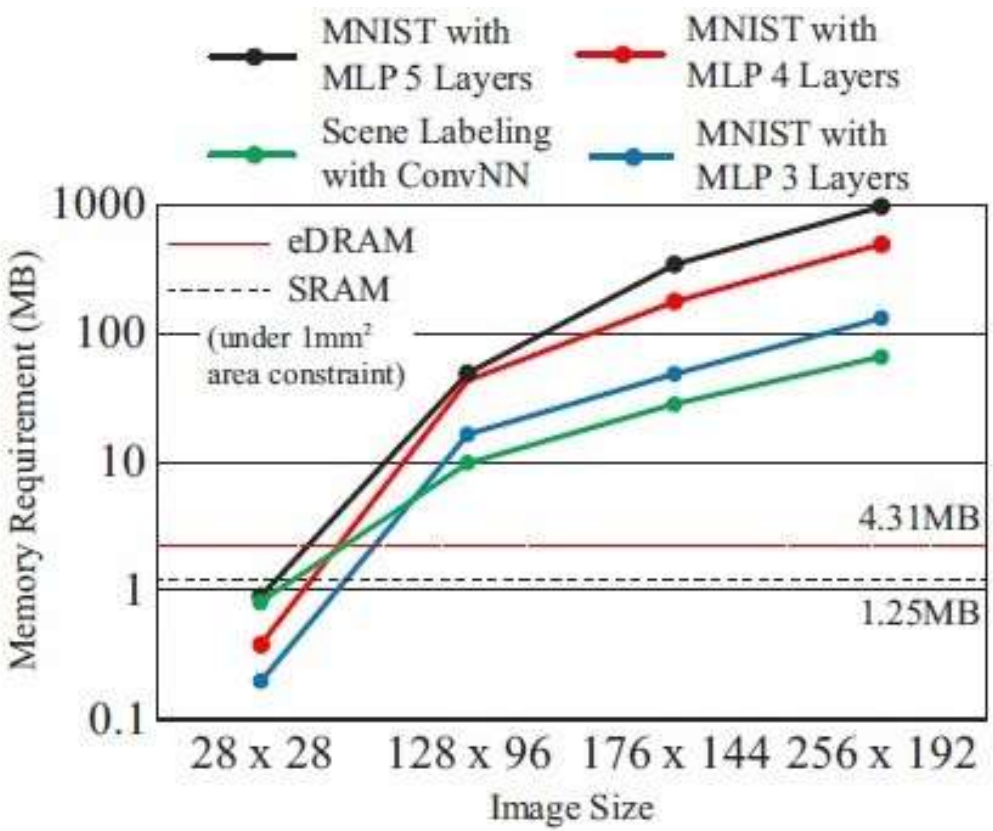


PIM example

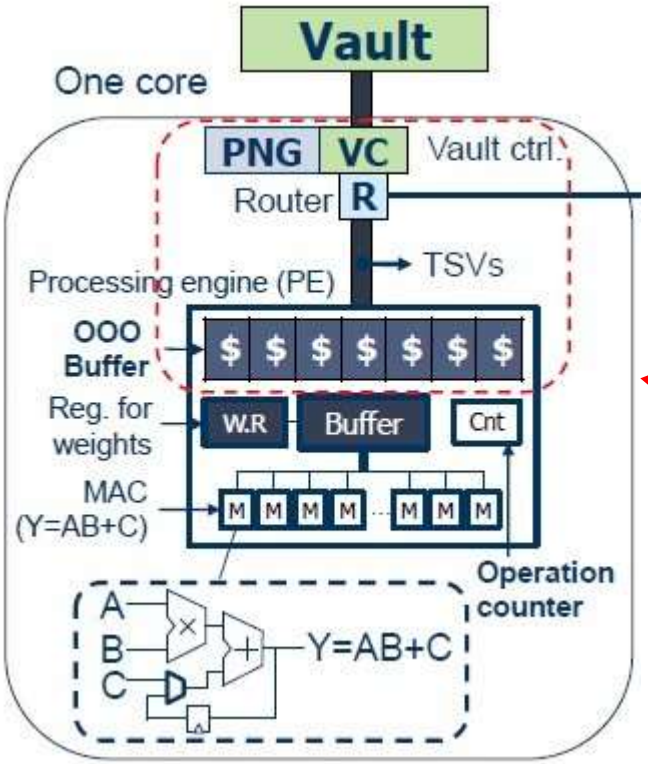
- Neurocube: A Programmable Digital Neuromorphic Architecture with High-Density 3D Memory
- In ISCA 2016
- Processing in memory for neural networks

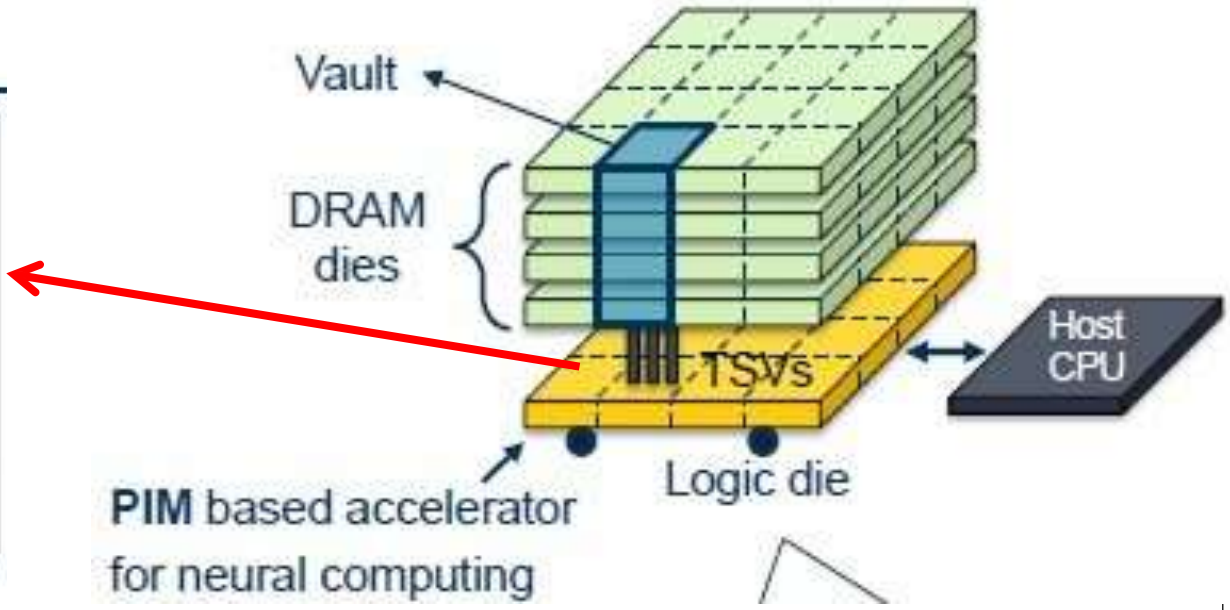
Neurocube motivation

- The growth in required memory capacity for two image processing application
- By far larger than on-chip memory capacity



Neurocube

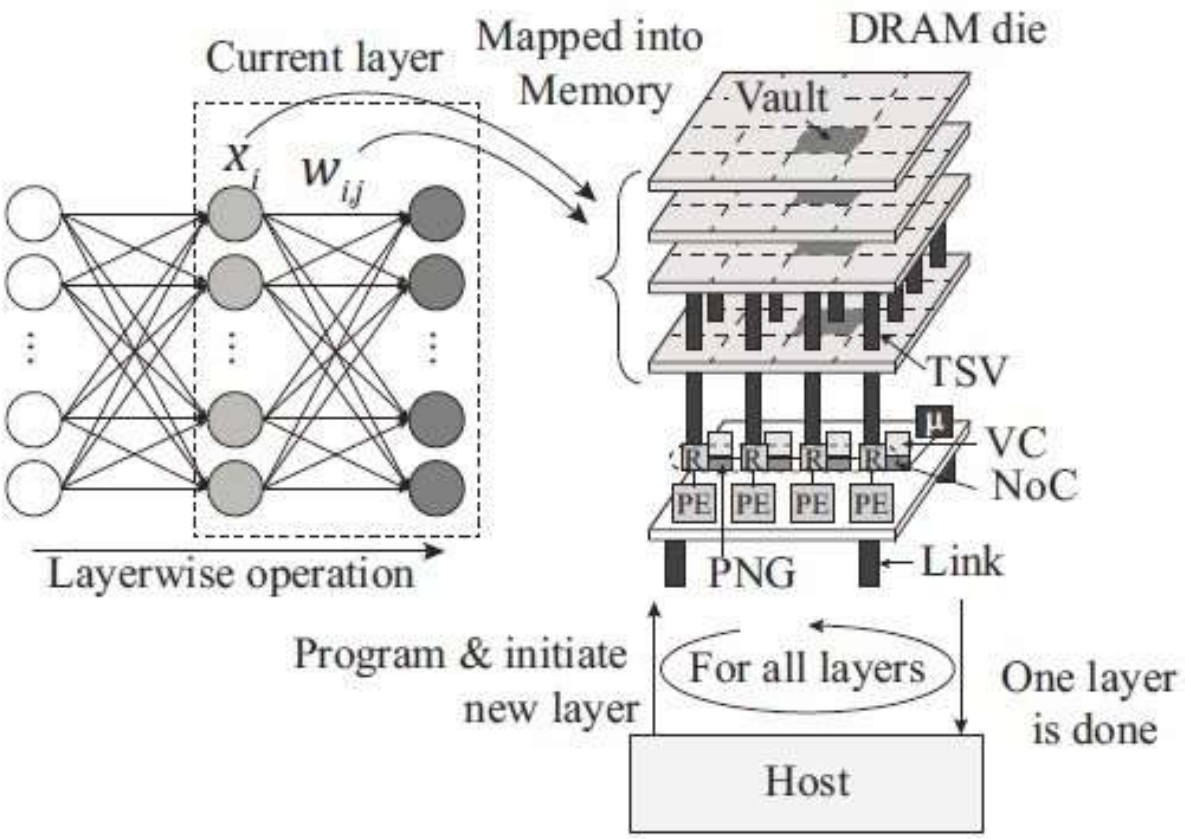




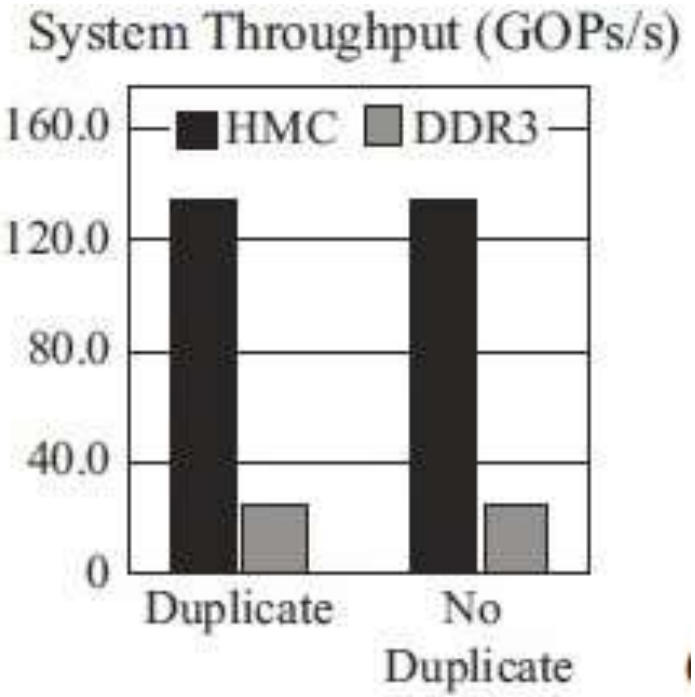
- Use neural network accelerator processing elements in each vault controller

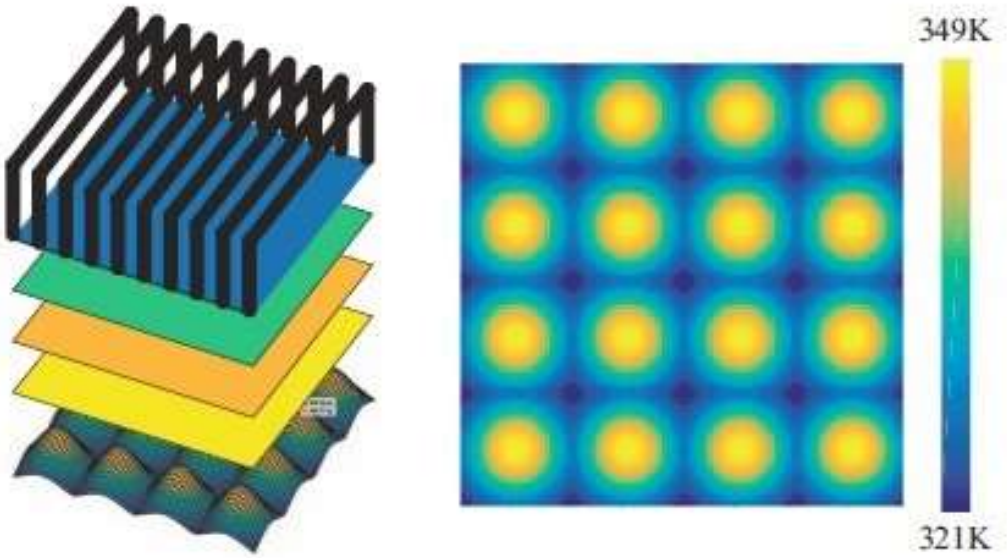
Neurocube

- Mapping data onto vaults
- Host processor performs mapping and PE initialization



Evaluation





Thermal analysis operating 5GHz (15nm FinFet)

- Both performance and temperature should be evaluated

Questions?

Thanks for your attention.