Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA

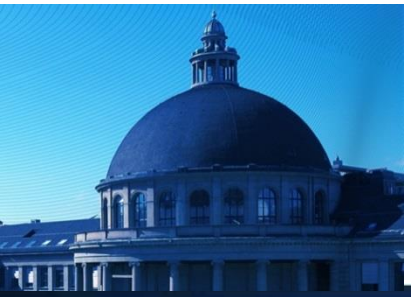
Design and Evaluation of a Processing-in-Memory Architecture for the Smart Memory Cube

Erfan Azarkhish(erfan.azarkhish@unibo.it)
Davide Rossi (davide.rossi@unibo.it)
Igor Loi (igor.loi@unibo.it)
Luca Benini (luca.benini@iis.ee.ethz.ch)

SAMSUNG ELECTRONICS

nano-tera.ch
YINS RTD
20NA21150939

Multitherman
GA n. 291125



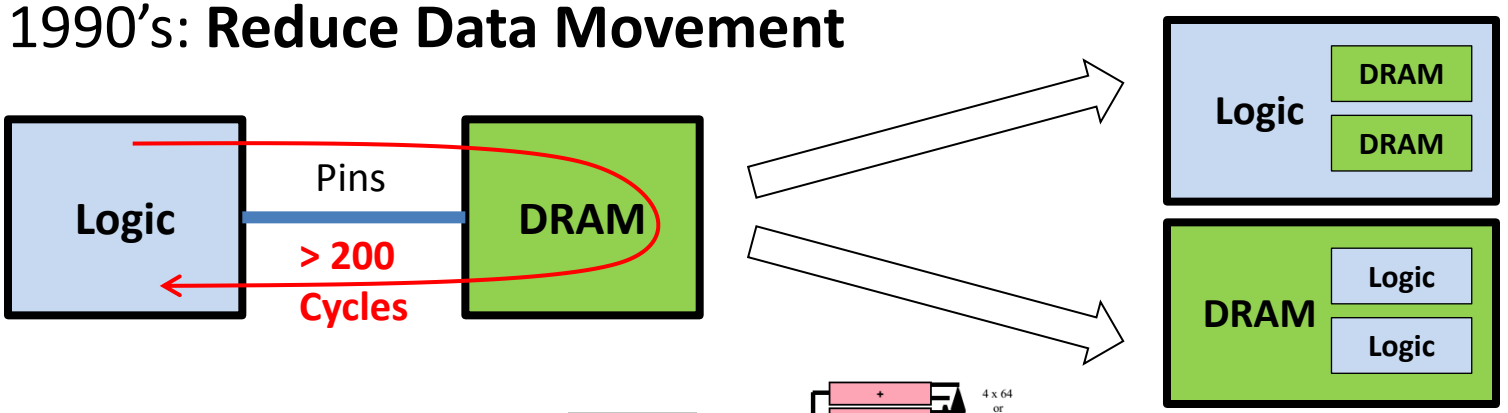
ARCS 2016 - ARCHITECTURE OF COMPUTING SYSTEMS
04-07 April 2016, Nuremberg, Germany

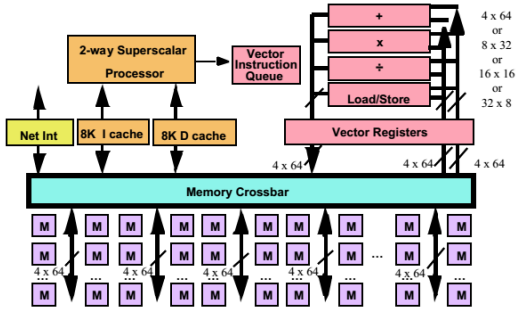
Outline

- Intro:
  - **Near Memory Computation**
  - **Smart Memory Cube (SMC)**
- Proposed **Processing-in-Memory (PIM)** Architecture for the SMC
- **Full-System** Simulation Results

Near Memory Computation (NMC)

Late 1990's: **Reduce Data Movement**





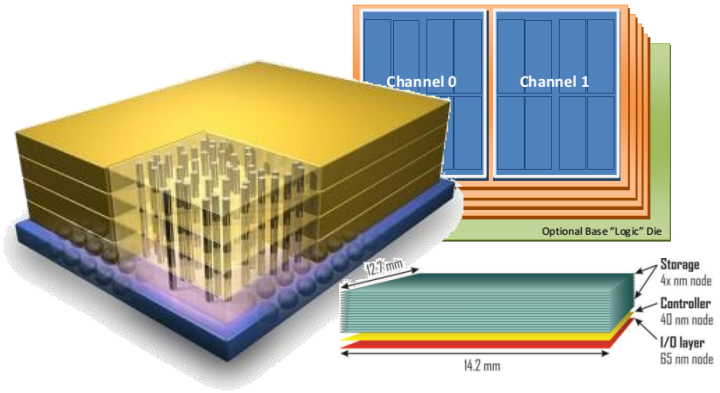
Intelligent RAM (IRAM)
[Patterson, MICRO'97]

Not successful **industrial** products

Incompatibility of **DRAM and logic** processes [Patterson, ICCD, 1997]

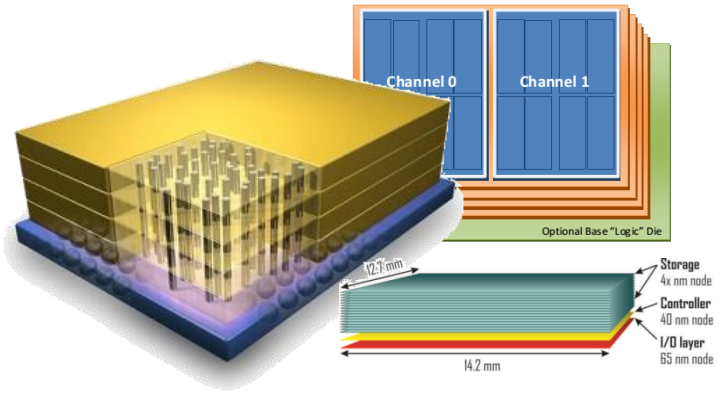
Heterogeneous 3D Integration

- Starting from 2011, this situation started to change: **Through-silicon-vias** (TSV)
  - **DRAM** and **logic** very **close to each other** but in their own processes
  - **Commercial** maturity → memory manufacturers
- Examples
  - **High Bandwidth Memory** (HBM)
  - **DiRAM4** 3D Memory
  - **Hybrid Memory Cube** (HMC)



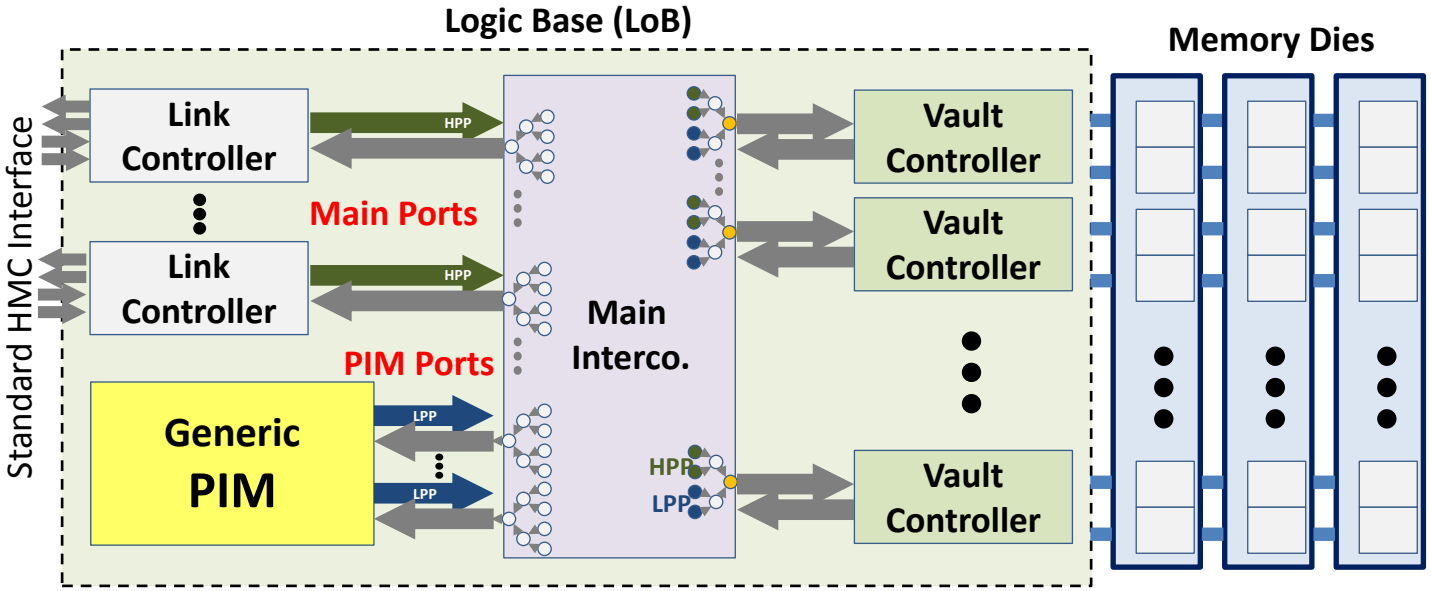
Heterogeneous 3D Integration

- Starting from 2011, this situation started to change: **Through-silicon-vias** (TSV)
  - **DRAM** and **logic** very **close to each other** but in their own processes
  - **Commercial** maturity → memory manufacturers
- Examples
  - **High Bandwidth Memory** (HBM)
  - **DiRAM4** 3D Memory
  - **Hybrid Memory Cube** (HMC)
    - + Backed up by HMC consortium
    - + Higher **flexibility** by **abstracting** away the details of DRAM control



➔ **Suitable for Near Memory Computation**

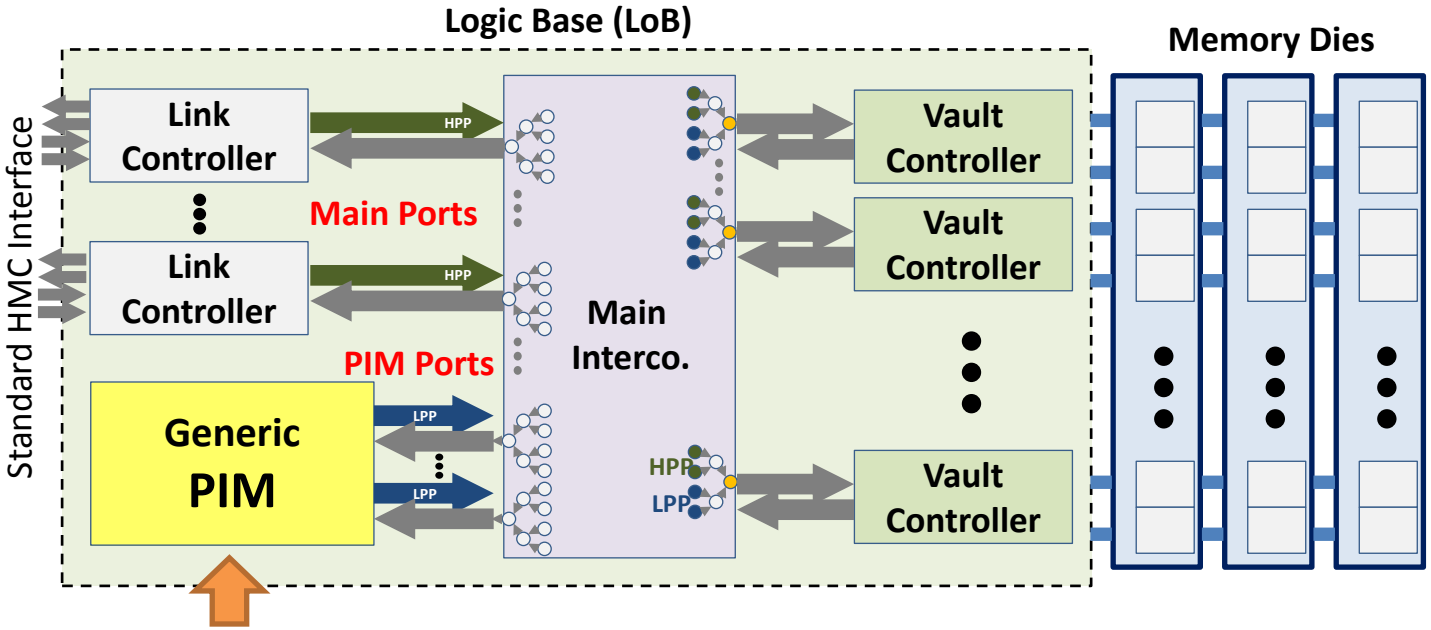
Smart Memory Cube (SMC)



+ A modular extension to the standard HMC
→ **fully compatible** with its IO interface
+ No new die is introduced in the stack

[E. Azarkhish, *et. al.* , "High performance AXI-4.0 based interconnect for extensible smart memory cubes," IEEE DATE, 2015]

Smart Memory Cube (SMC)



+ PIM has a **global visibility** of the whole memory space
  + Modular and scalable
  + No concern about DRAM process

Cycle Accurate Model

Traffic-based Analysis

[E. Azarkhish, *et. al.* , "High performance AXI-4.0 based interconnect for extensible smart memory cubes," IEEE DATE, 2015]

In this work: we design a PIM processor

Motivations for Near Memory Computation

Literature:

1. Latency Reduction: Reduction in data movement
2. Higher bandwidth: TSVs instead of Pins

Observation: **HMC's serial links** can deliver **all its internal bandwidth** to the outside
(4 Links) x (32 Lanes) x (30Gbps) > 320GB/s

Motivations for Near Memory Computation

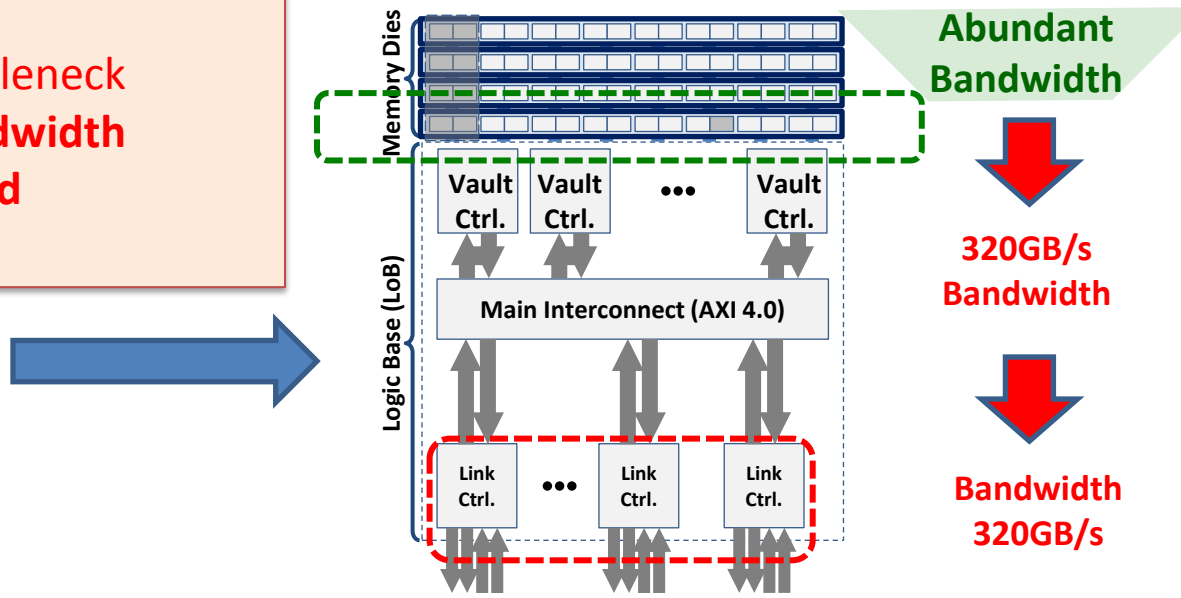
Literature:

1. Latency Reduction: Reduction in data movement
2. ~~Higher bandwidth: TSVs instead of Pins~~

Observation: **HMC's serial links** can deliver **all its internal bandwidth** to the outside
(4 Links) x (32 Lanes) x (30Gbps) > 320GB/s

Inside HMC:
- **TSV interface** is the main bottleneck
- **Logic-base** has the **same bandwidth** available as the **external world**
- **Lower latency**

What are PIM's benefits
- Located on the Logic Base
- Performance? Energy?



State of the Art (2013-2015)

SoA	PIM Model	Platform	Cache on PIM	Cache Coherence	Memory Management	Application	Energy Saving	Perf. Gain
NDA: [HPCA'15]	CGRA	gem5	No	Uncacheable	Segmentation with no paging	Big Data (MapReduce)	46%	1.6X
AMD + UNT: [Euro-Par'14]	Cortex-A5 x 64	gem5	L1I, L1D, L2	Software flush	Contiguous preallocated	Big Data (MapReduce)	23%	1.1X
Sandia.GOV [CO-HPC'14]	Light GP Cores x 16	SST (Instruction Trace)	L1I, L1D, L2	Yes - Hardware	Preallocation	Scientific: Sparse Linear Algebra, …	-	~2X
TOP-PIM: [HPDC'14]	PIM= CPU+GPU	Analytical	L1, L2	Gathered Statistics	Preallocated	Graph, HPC, GPGPU	85%	7%
SNU+CMU [ISCA'15]	Low-level operations	In-house	No	Yes - Hardware	Full Virtual Memory	Big Data	1.6X	20%
AMC: [CF'15]	Vector Processors	Mambo simulator	No	Yes - Hardware	Full Virtual Memory + Allocation	Dense Matrix Operations	-	Up to 5X
Utah: [ISPASS'14]	Cortex-A5 x 1000	SIMICS Trace-based	L1	-	-	Big Data (MapReduce)	18X	15X
LiM: [3DIC'13]	App. Specific	Sniper Trace-based	-	-	Preallocation	Dense Matrix, BLAS, FFT	>100X	>100X
SNU+CMU [ISCA'15]	In-order cores x 512	In-house	L1, L2, Prefetchers	Uncacheable	Segmentation	Page Rank	87%	10X

State of the Art (2013-2015)

SoA	PIM Model	Platform	Cache on PIM	Cache Coherence	Memory Management	Application	Energy Saving	Perf. Gain
NDA: [HPCA'15]	CGRA	gem5	No	Uncacheable	Segmentation with no paging	Big Data (MapReduce)	46%	1.6X
AMD + UNT: [Euro-Par'14]	Cortex-A5 x 64	gem5	L1I, L1D, L2	Software flush	Contiguous preallocated	Big Data (MapReduce)	23%	1.1X
Sandi [CO-H								~2X
TOP- [HPD								7%
SNU- [ISC								20%
AN [CF'15]	Processors	simulator	No	Hardware	Memory + Allocation	Operations	-	Up to 5X
Utah: [ISPASS'14]	Cortex-A5 x 1000	SIMICS Trace-based	L1	-	-	Big Data (MapReduce)	18X	15X
LiM: [3DIC'13]	App. Specific	Sniper Trace-based	-	-	Preallocation	Dense Matrix, BLAS, FFT	>100X	>100X
SNU+CMU [ISCA'15]	In-order cores x 512	In-house	L1, L2, Prefetchers	Uncacheable	Segmentation	Page Rank	87%	10X

Observations:

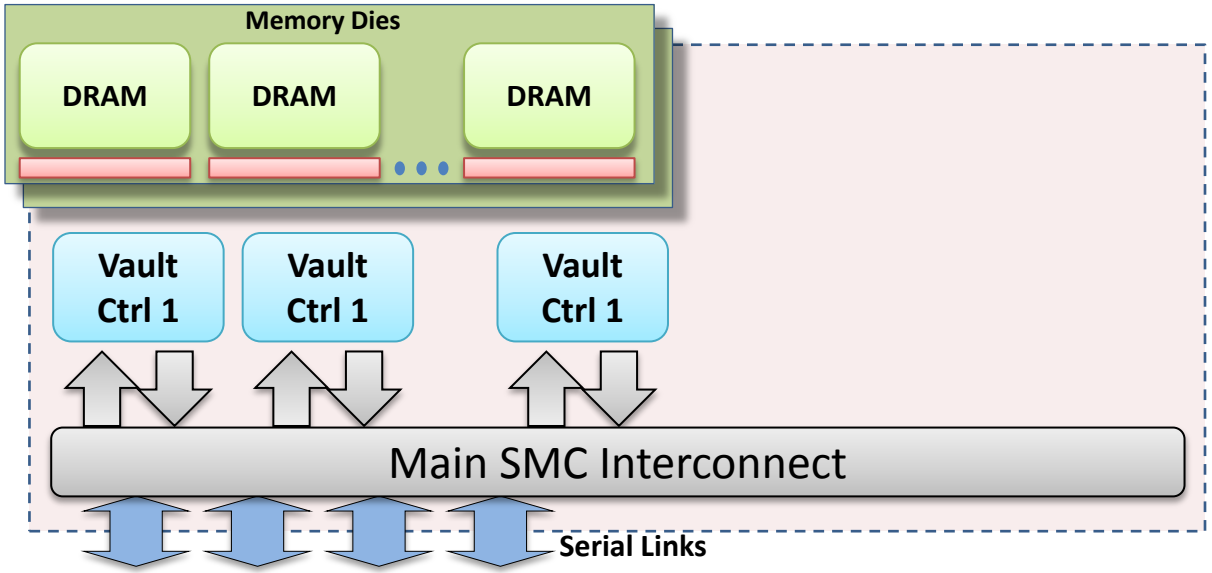
1. **Open Loop** (Trace-driven) simulation may give very inaccurate results for **PIM**
2. **Full System simulation** is desired (OS and all software and hardware layers)
3. **For 1 memory cube**: Less than 2X improvements are reported
4. With **networks of multiple memory cubes** are used ➔ ~10X improvement
   ➔ High end applications

Our Contributions

- **Full-system simulation** environment for SMC
- Looking in depth into the offload model, including **driver** and **OS** effects and constraints
- Address virtual memory issues
  - Evaluate their overheads
- Assessing energy and performance advantages in a **minimalistic setting**:
  - (Small PIM configuration)
  - Minimum cost and power overheads

Proposed PIM Architecture

The Smart Memory Cube (SMC)

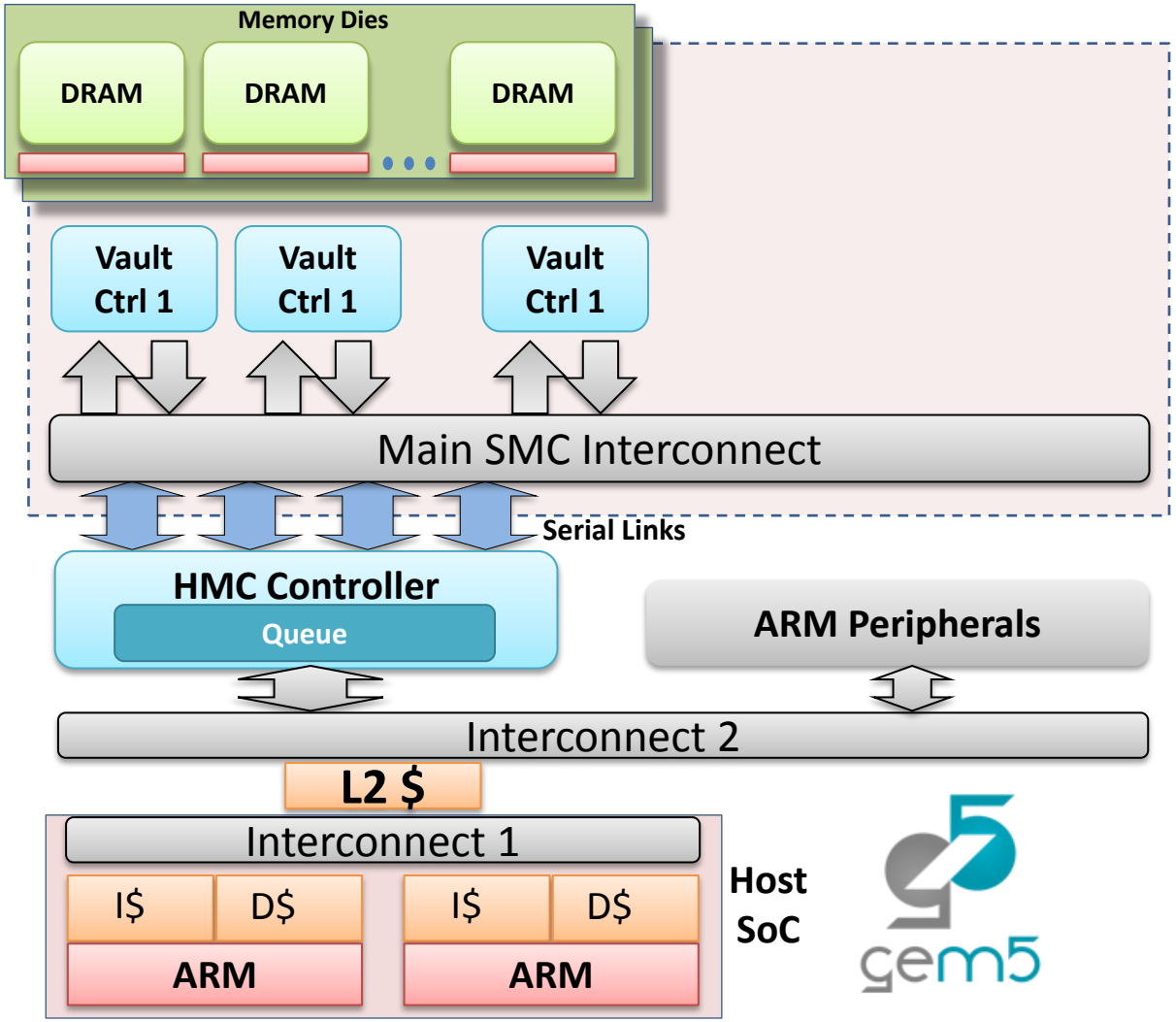


Proposed PIM Architecture

The Smart Memory Cube (SMC)

Host Side Controller

Complete ARM Host

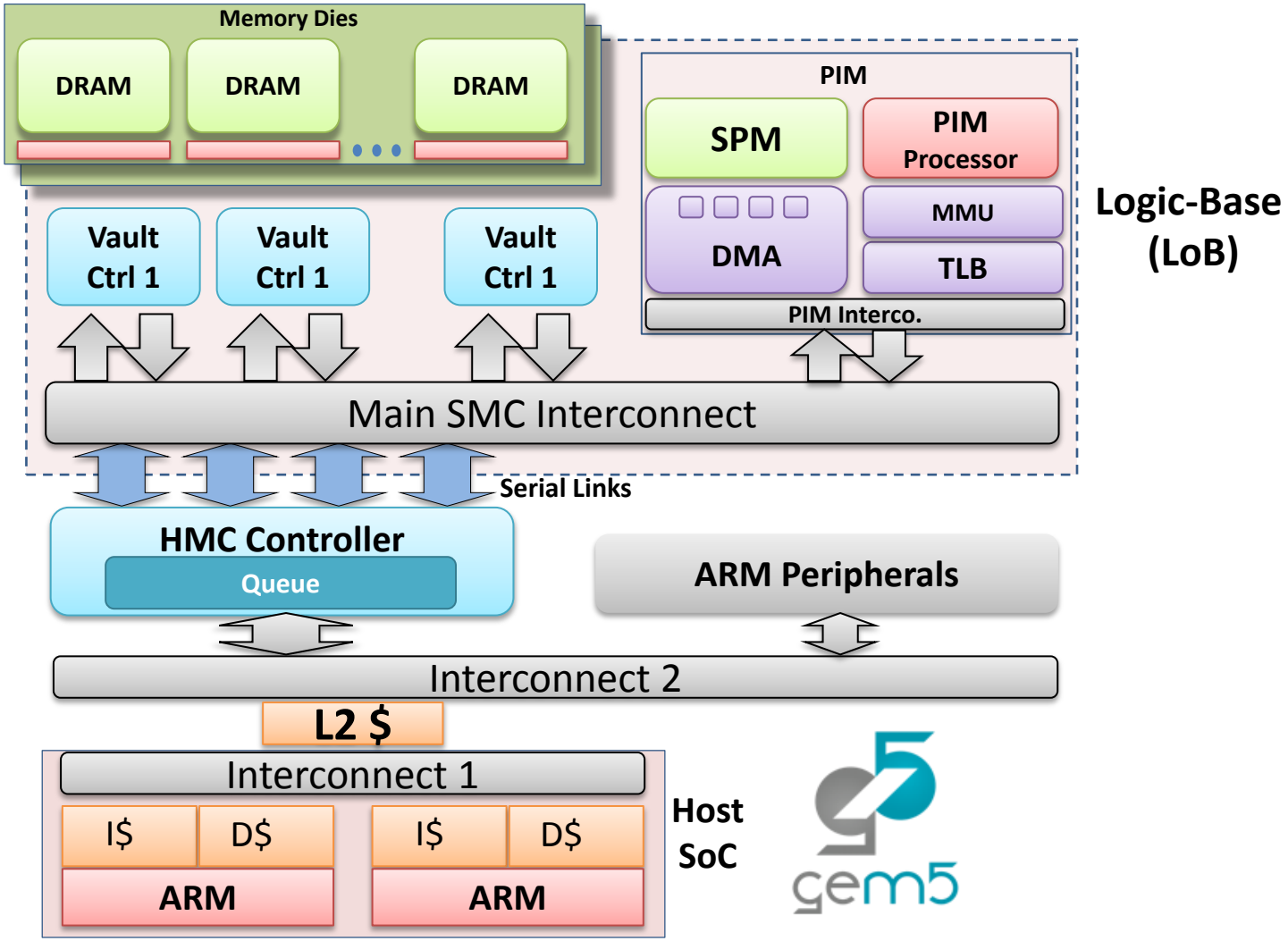


Proposed PIM Architecture

The Smart Memory Cube (SMC)

Host Side Controller

Complete ARM Host



Hardware Features

- **Maintain the standard interface of HMC**
  - **Memory mapped** communications
  - **No change in the packet protocol**
- **Flexible execution of different kernels**
  - **ELF Binary offloading**
- Zero-copy **virtual memory** support
  - **TLB** + Page Table
- **Bulk data transfer**:
  - Virtually addressed **DMA engine + Scratchpad Memory**
- **Extend atomic HMC Commands**
  - Local computations inside DRAM dies

```
for ( unsigned r=0; r<NODES; r++ )
{
    if ( nodes[r].teenager )
        for ( unsigned c=0; c< NODES; c++ )
        {
            if ( adj[r][c] != NC )
                nodes[c].followers++;
        }
}
```

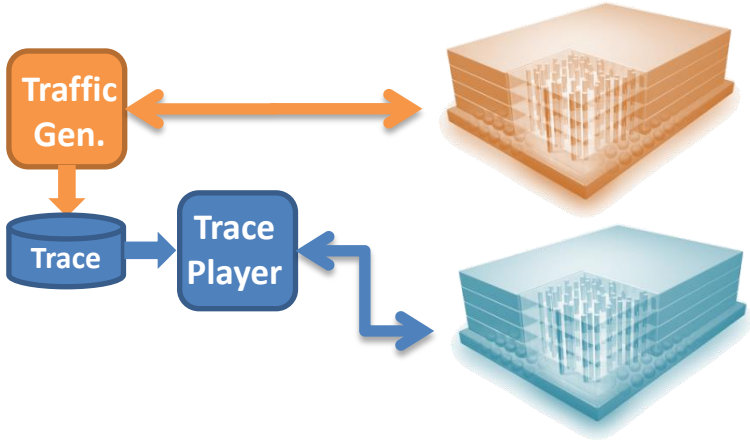
2 DRAM Accesses

```
for ( unsigned r=0; r<NODES; r++ )
{
    if ( nodes[r].teenager )
        for ( unsigned c=0; c< NODES; c++ )
        {
            if ( adj[r][c] != NC )
                HMC_ATOMIC_INC(nodes[c].followers);
        }
}
```

1 DRAM Accesses

Modeling Methodology

- **High level HMC** model in *gem5* *(General Memory System)*
- **Calibrated and verified** based on our **CA model of HMC**



High-level SMC Model

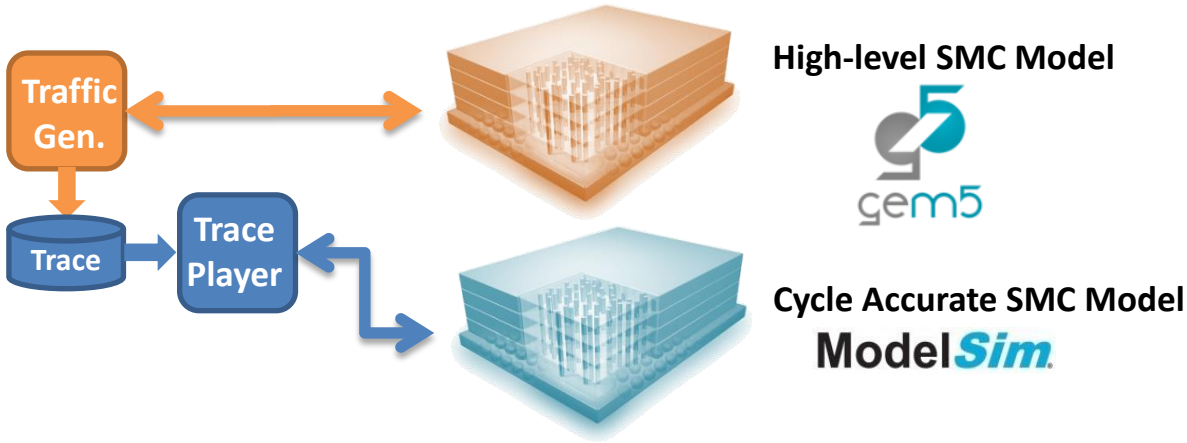
gem5

Cycle Accurate SMC Model [E. Azarkhish, *et. al.* IEEE DATE, 2015]

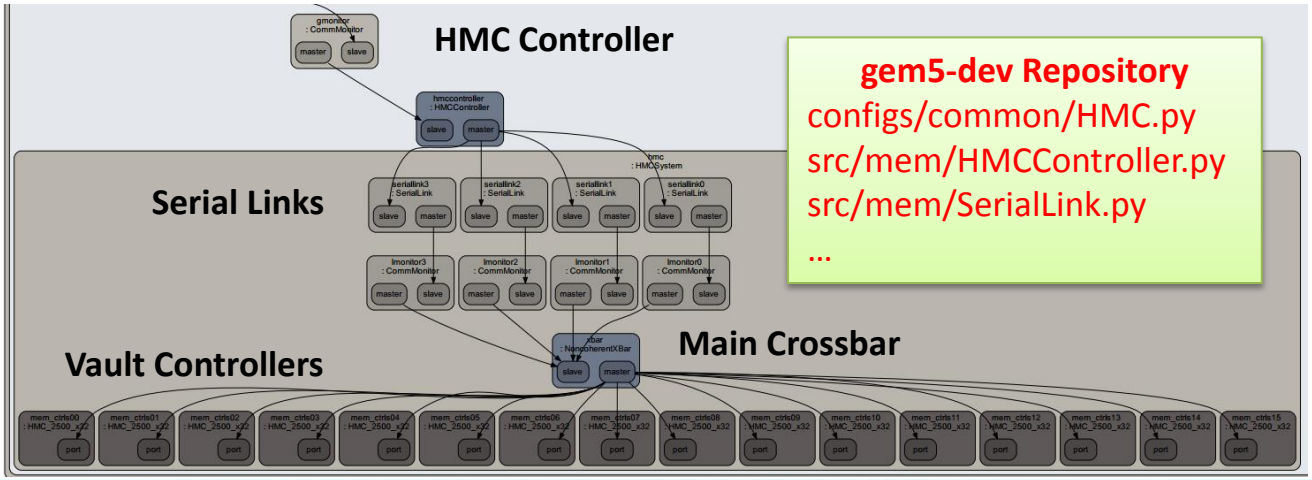
ModelSim

Modeling Methodology

- **High level HMC** model in *gem5*
- **Calibrated and verified** based on our **CA model of HMC**



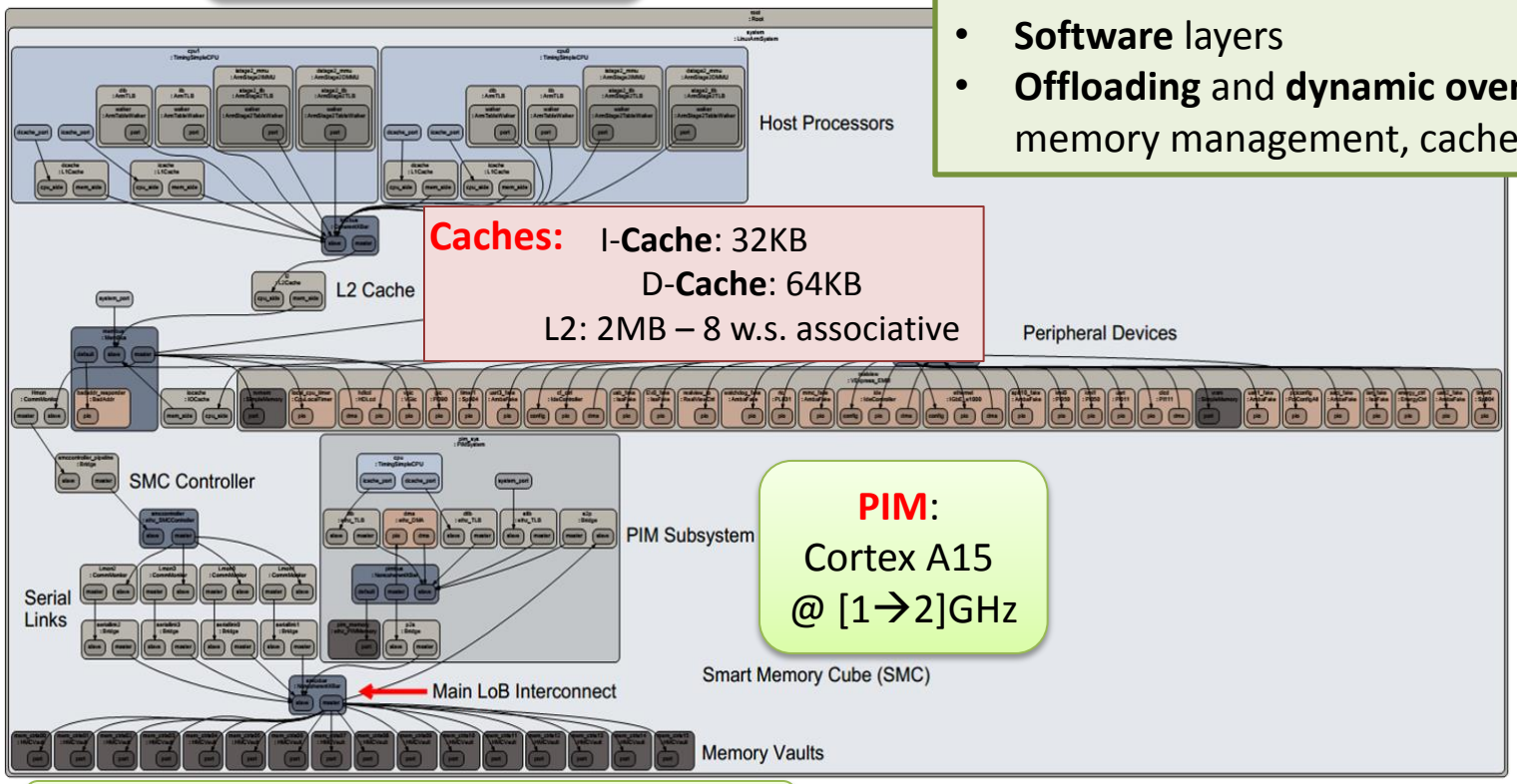
- Pushed to http://reviews.gem5.org/r/2986/



Complete model in gem5

Host:
2 x Cortex A15 @2GHz

Full System Simulation:
- **Linux OS**
- **Software** layers
- **Offloading** and **dynamic overheads**: memory management, cache coherence, …



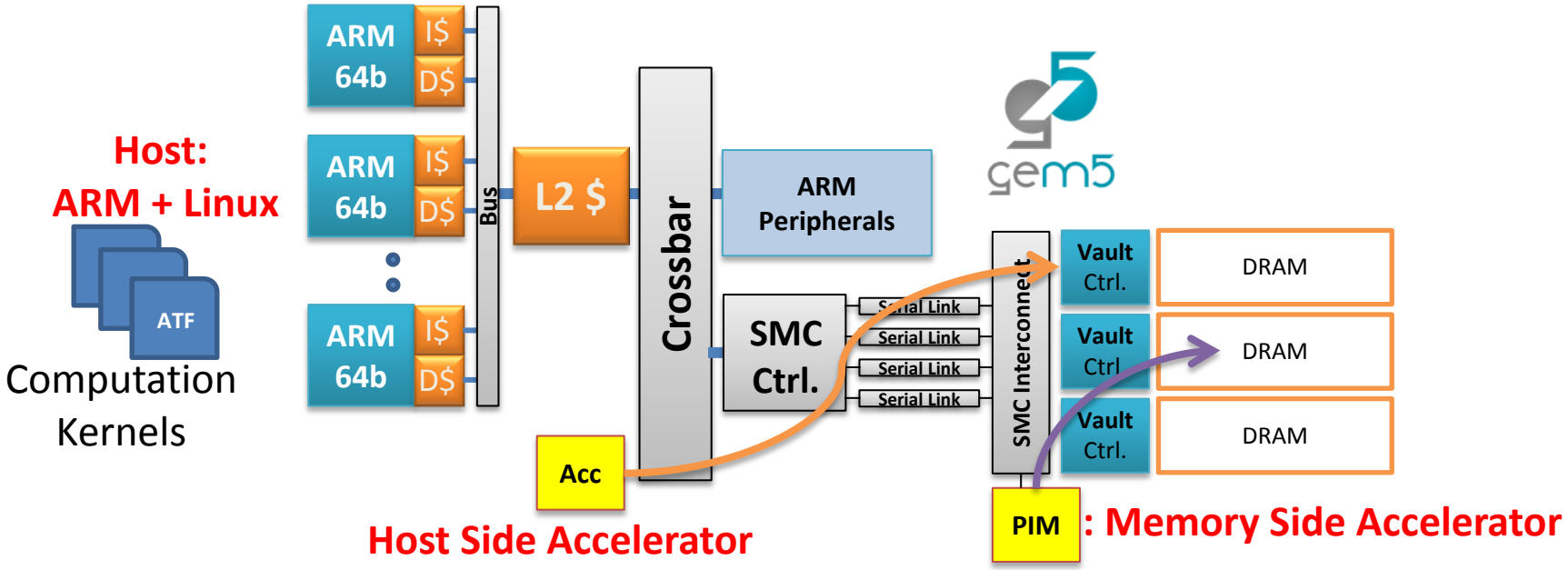
Caches: I-**Cache**: 32KB
D-**Cache**: 64KB
L2: 2MB – 8 w.s. associative

PIM:
Cortex A15
@ [1→2]GHz

SMC: 512MB – 16 Vaults – 4 Mem Dies.

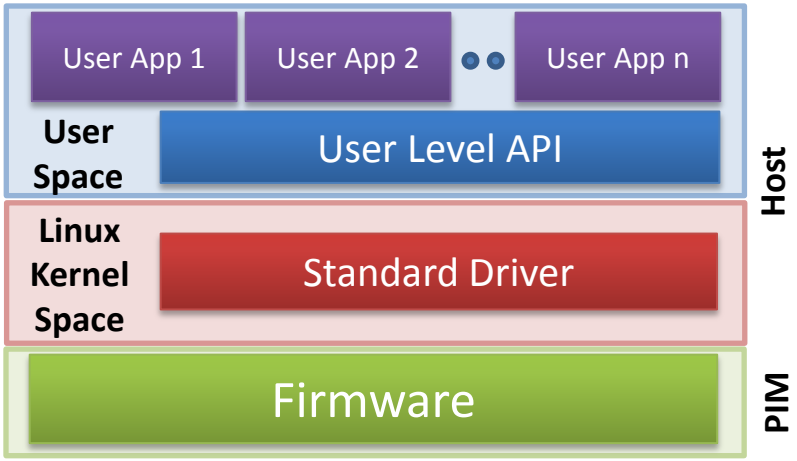
Evaluated Scenarios

1. **Kernel Offloading:**
   - ELF Binary offloading
2. **Task Offloading:** (User Data Structures)
   - Build page table – flush the caches – pass the pointers
3. **Actual Execution**

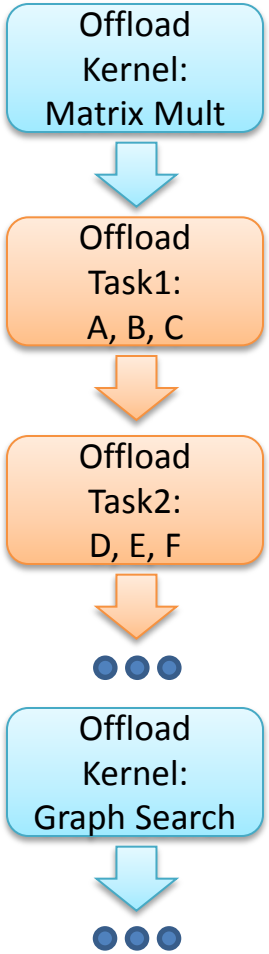


Programming Model

- User Level API (User Space)
  - Abstract away the details
  - Multiple user-level Apps
- Standard Driver (Kernel Space)
  - Adopted from MALI GPU
- Firmware & Memory mapped registers



Software Stack



Benchmarks

Large-Scale **Graph Processing** Kernels
➔ **Latency Sensitive**

- Average Teenage Follower (**ATF**)
- Google's Page Rank (**PR**)
- Breadth First Search (**BFS**)
- Bellman Ford Shortest Path (**BF**)

[Ahn et. al., ISCA'15]

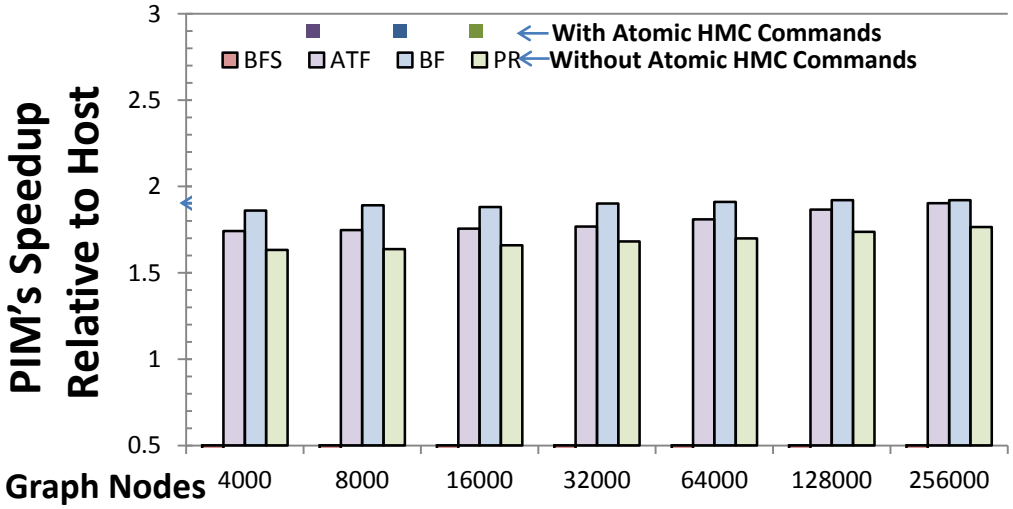
Large Graphs (~500K Nodes)

```
for ( ulong_t i=0; i<NODES; i++ )
    {
      nodes[i].page_rank = 1.0 / NODES;
      nodes[i].next_rank = 0.15 / NODES;
    }
    ulong_t count = 0;
    float diff = 0.0;
    do {
      for ( ulong_t i=0; i<NODES; i++ )
      {
        float delta = 0.85 * nodes[i].page_rank / nodes[i].out_degree;
        for ( ulong_t j=0; j<nodes[i].out_degree; j++ ) // for node.successors
            nodes[i].successors[j]->next_rank += delta;
      }
      diff = 0.0;
      for ( ulong_t i=0; i<NODES; i++ )
      {
        diff += fabsf(nodes[i].next_rank - nodes[i].page_rank);
        nodes[i].page_rank = nodes[i].next_rank;
        nodes[i].next_rank = 0.15 / NODES;
      }
    } while (++count < PAGERANK_MAX_ITERATIONS && diff >
PAGERANK_MAX_ERROR);
```

Offload to PIM for acceleration
LIL Representation

- Compact and scalable

Performance Results

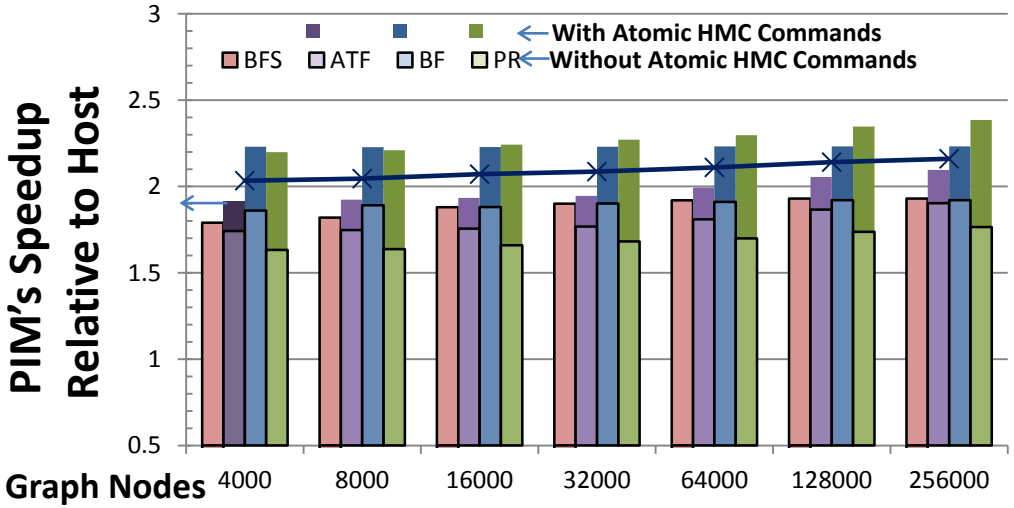


Average Speedup of PIM:

- **1.7X compared to the host SoC**
- **1.3X compared to host-side accelerator**

Including all offloading overheads

Performance Results



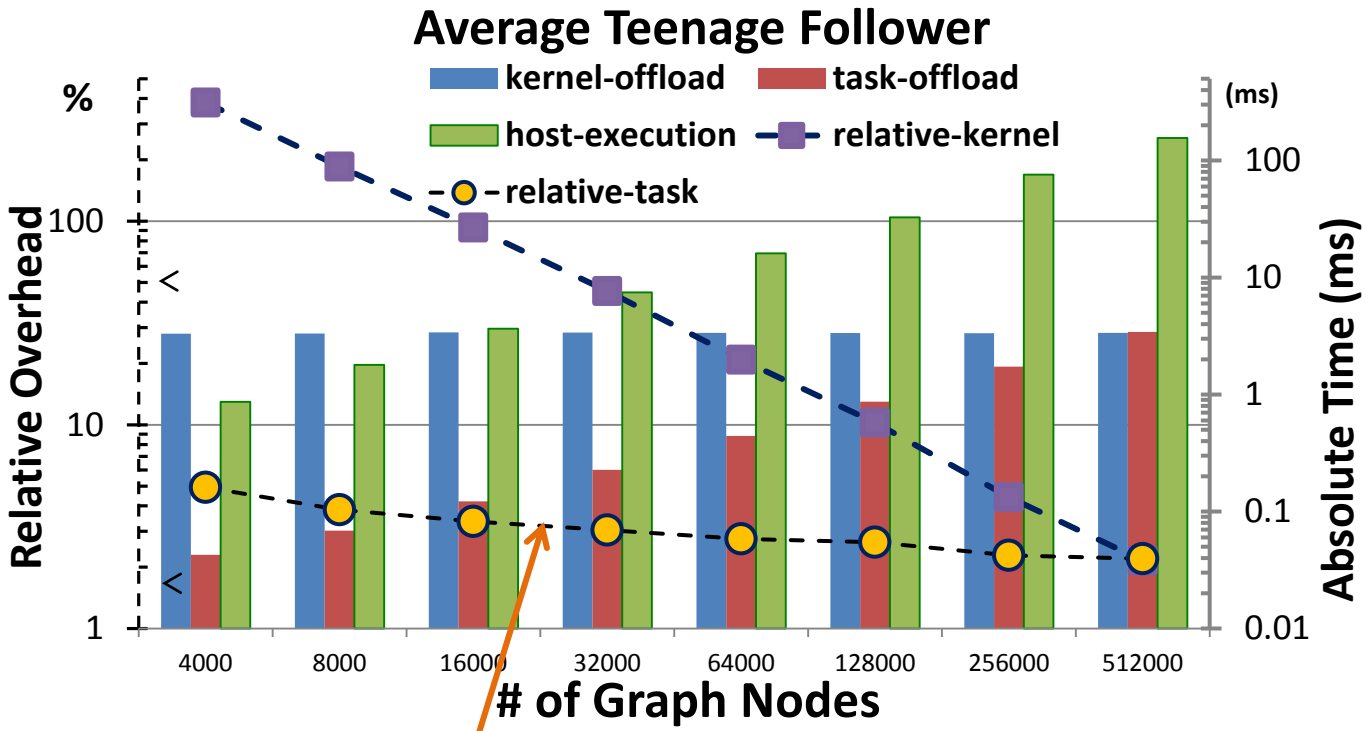
Average Speedup of PIM:
- **1.7X compared to the host SoC**
- **1.3X compared to host-side accelerator**

➔ If we extend the **Atomic HMC Operations**:

atomic-min, **atomic-increment**, and **atomic-add-immediate**

Average Speedup of PIM:
- **Graph benchmarks:**
  - **2X compared with the host**
  - **1.5X compared to the host-side accelerator**

Offloading Overheads



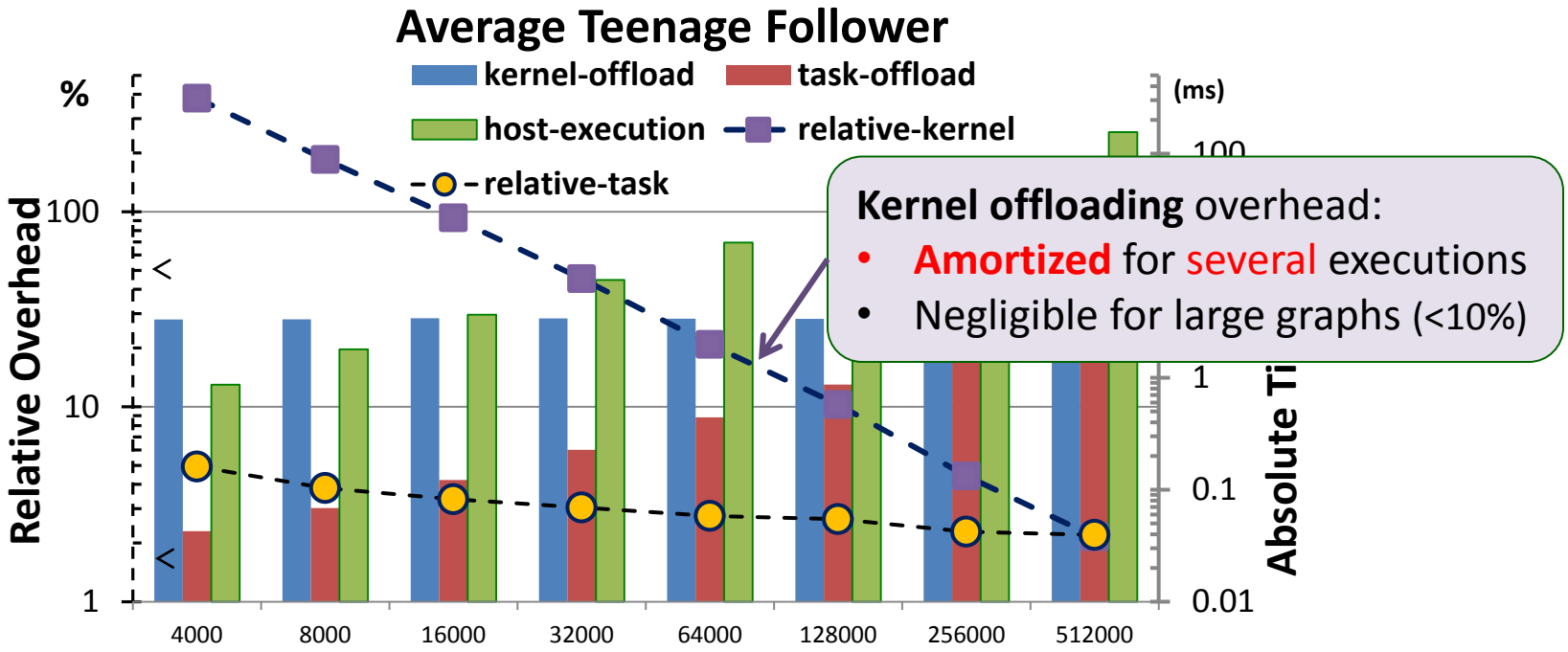
Task offloading overhead

- Always below 5%
- For 500K nodes: less than 2%

API & Driver:

- VA → PA translation
- Page Table
- Cache Flush
- Send Pointers

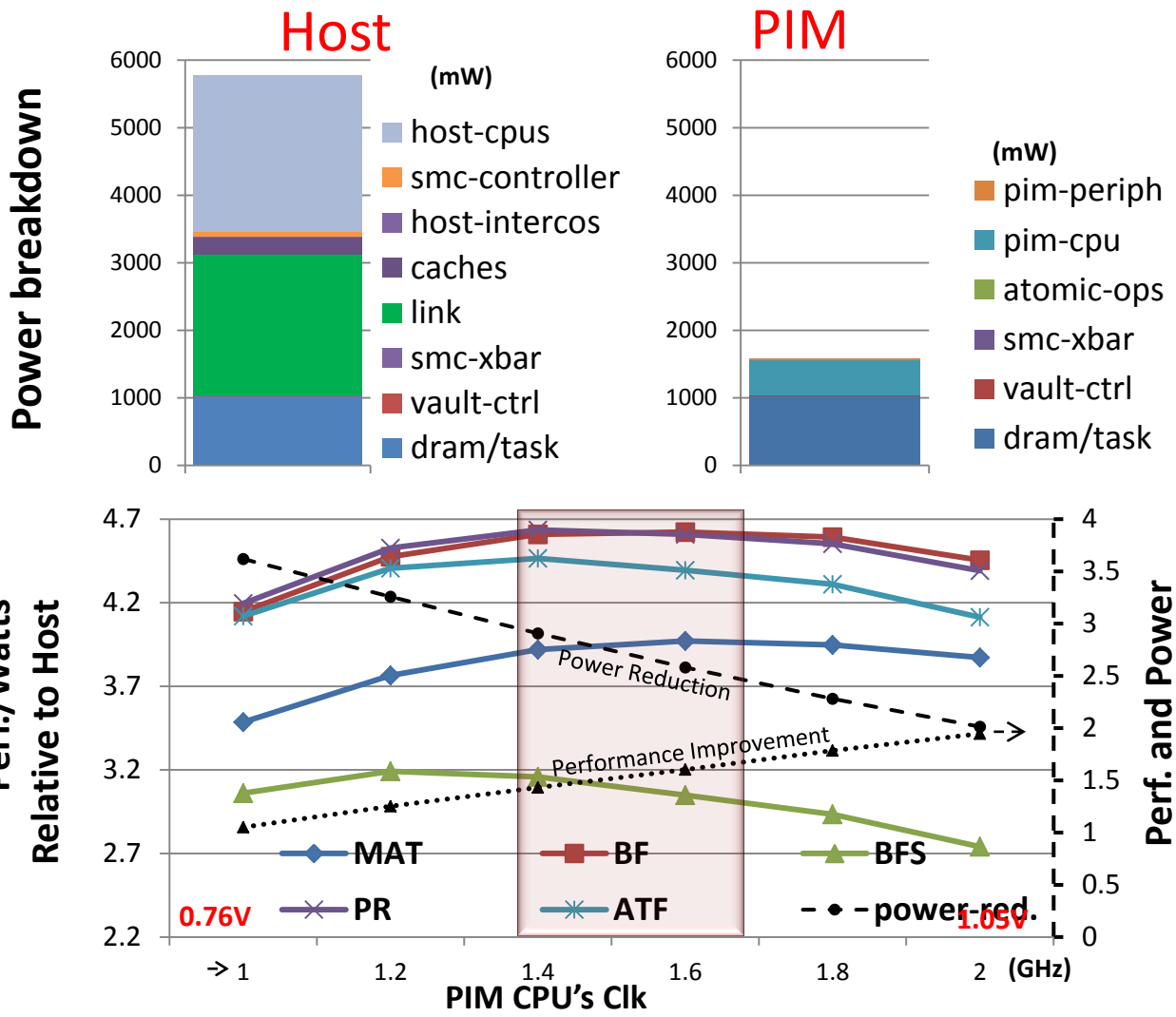
Offloading Overheads



System Power Estimation

- **Logic Synthesis:** Interconnects, TLB, DMA Engine:
- **CACTI**: Caches and SPM
- **DRAMPower:** DRAM devices:
- **Excel Sheets:**
  - **Serial Links**: [S. Lloyd, Computer 2015] [P. Rosenfeld, 2014], [J. Jeddeloh, VLSIT 2012]
  - **Link Power Gating**: [J. Ahn, TVLSI 2015]
  - **Vault Controllers**: [B. Boroujerdian, Berkley, 2012]
  - **HMC Controller**: [M. Schaffner, DATE 2015]
  - **ARM Processors**: [B.M. Tudor, SIGMETRICS 2013]
  - **Atomic** Operations: [A. Aminot, ARCS 2015]
- **Background Power** was ignored:
  - Peripheral devices, secondary storage, cooling mechanism, system's clocking,
  - Unused DRAM cells (Memory Usage < 100MB)

Power Efficiency



Power Reduction: 70%
(Serial Links, Host Caches)

Optimum Perf./Watts
Host: 2GHz
PIM: 1.5GHz
3X Energy efficiency impr.
1.5X Perf Improvement
50% energy reduction

Summary

- **High-level HMC model** in gem5 model **with verified accuracy**
- **Full System Simulation**
- **Zero-copy virtual address** support

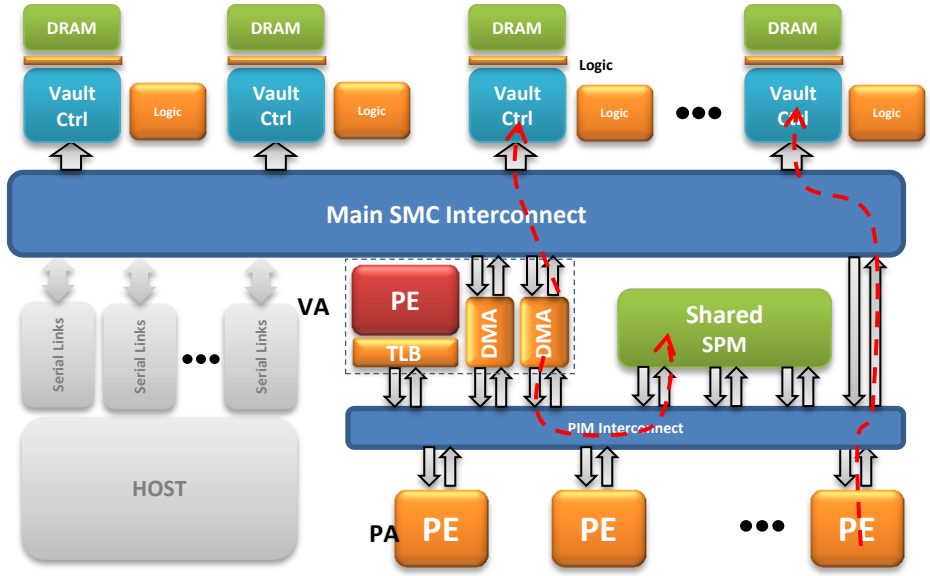
  ➔ Negligible overheads
- **In a minimalistic setting: Single Core PIM + DMA + …**
  - **Performance**:
    - up to **2X performance improvement** in comparison with the **host processor**
    - **1.5X performance** compared with host-side **accelerator**
  - **Energy**: (Same performance)
    - **70%** energy reduction (host SoC)
    - **55%** energy reduction (host side accelerator)
- PIM requires **less buffering** compared to the **host side accelerator**
  - **To maintain the same bandwidth**
    - Due to lower access latency

  ➔ Further cost and energy optimizations

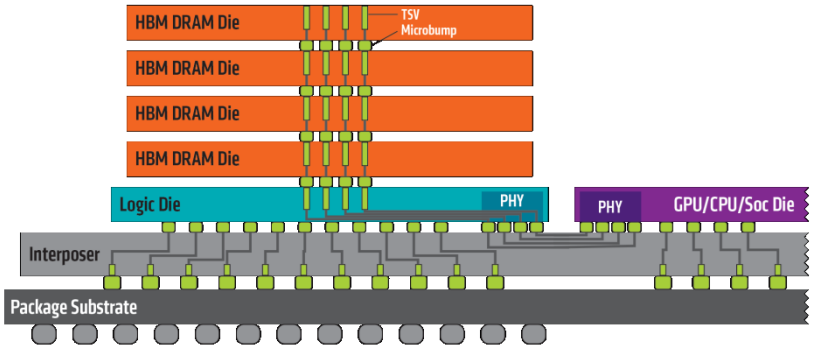
Little's Law: $L = \lambda W$

Ongoing works

- Cluster of **low-power PIM processors**
  - Parallelized kernels



- Applicability of our method to High-bandwidth Memory (HBM)
  - No abstracted interface, but
  - Less power overheads
  - Deal with DRAM commands



Source: www.amd.com

Thank you, questions?

(1) E. Azarkhish, D. Rossi, I. Loi, and L. Benini, “High performance AXI-4.0 based interconnect for extensible smart memory cubes,” IEEE DATE, 2015

(2) J. Jeddeloh and B. Keeth, Hybrid memory cube new DRAM architecture increases density and performance in VLSI Technology (VLSIT), 2012 Symposium on, June 2012, pp. 87–88.

(3) P. Rosenfeld, Performance exploration of the hybrid memory cube Ph.D. dissertation, univ. of Maryland, 2014.

(4) G. Kim, J. Kim, J. H. Ahn, and J. Kim, Memory-centric system interconnect design with hybrid memory cubes, in Parallel Architectures and Compilation Techniques (PACT), 2013 22nd International Conference on, Sept 2013, pp. 145–155

(5) A. Farmahini-Farahani, et. al., "NDA: Near-DRAM acceleration architecture leveraging commodity DRAM devices and standard memory modules," HPCA 2015

(6) G. Stelle, *et. al.*, “Using a Complementary Emulation-Simulation Co-Design Approach to Assess Application Readiness for Processing-in-Memory Systems”, CO-HPC'14

(7) Z. Sura, et. al., “Data access optimization in a processing-in-memory system”, CF 2015

(8) Q. Zhu, et. al., “A 3D-Stacked Logic-in-Memory Accelerator for Application-Specific Data Intensive Computing”, 3DIC 2013

(9) D. Zhang, et. al., “TOP-PIM: Throughput-Oriented Programmable Processing in Memory”, HPDC 2014

(10) J. Ahn, *et. al.* “PIM-Enabled Instructions: A Low-Overhead, Locality-Aware Processing-in-Memory Architecture”, ISCA 2015

(11) J. Ahn, *et. al.*, “A Scalable Processing-in-Memory Accelerator for Parallel Graph Processing”, ISCA 2015