Kindle: A Comprehensive Framework for Exploring OS-Architecture Interplay in Hybrid Memory Systems

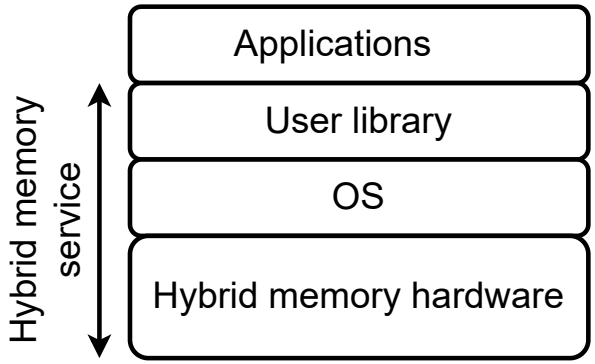
IISWC 2024

Arun KP Debadatta Mishra

Indian Institute of Technology, Kanpur

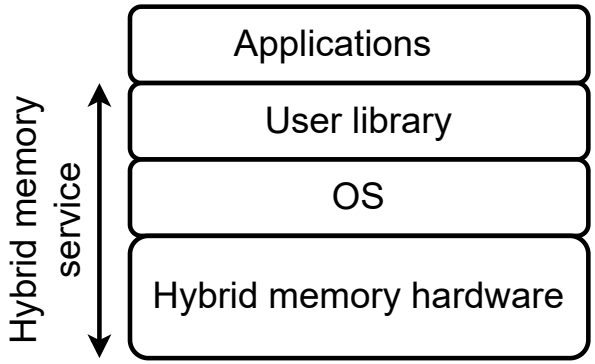
Hybrid Memory System

- Hybrid memory provides benefits of both DRAM and NVM.



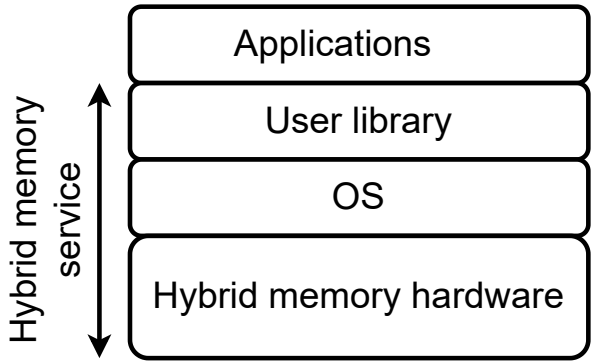
Hybrid Memory System

- Hybrid memory provides benefits of both DRAM and NVM.
- Applications can take advantage of hybrid memory using services at different levels.



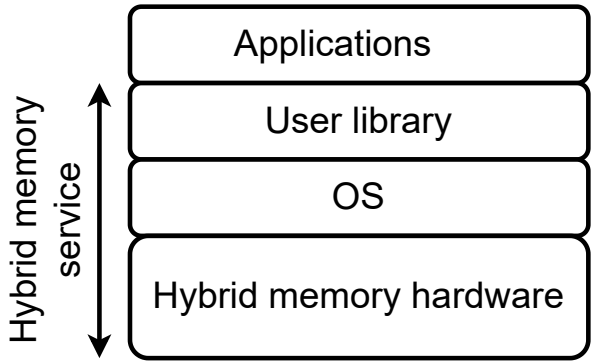
Hybrid Memory System

- Hybrid memory provides benefits of both DRAM and NVM.
- Applications can take advantage of hybrid memory using services at different levels.
- Hybrid memory service example $\rightarrow$ page migration, data persistence, etc.



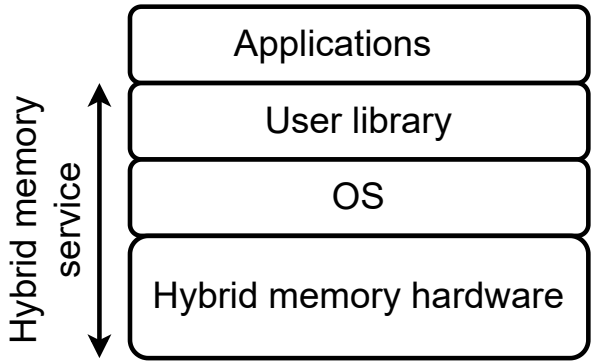
Hybrid Memory System

- Hybrid memory service may span multiple layers.



Hybrid Memory System

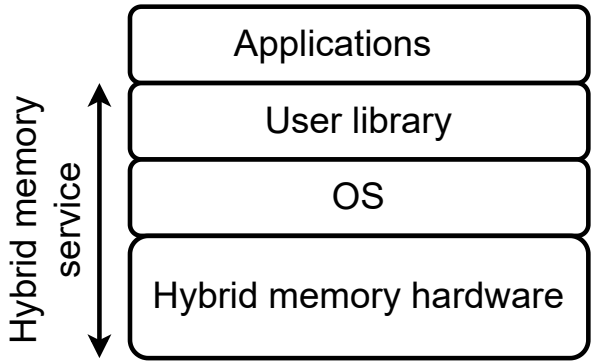
- Hybrid memory service may span multiple layers.
- HSCC $\rightarrow$ page migration (OS + hardware)



HSCC: Hardware/software cooperative caching for hybrid DRAM/NVM memory architectures

Hybrid Memory System

- Hybrid memory service may span multiple layers.
- HSCC $\rightarrow$ page migration (OS + hardware)
- SSP $\rightarrow$ data persistence (OS + hardware)

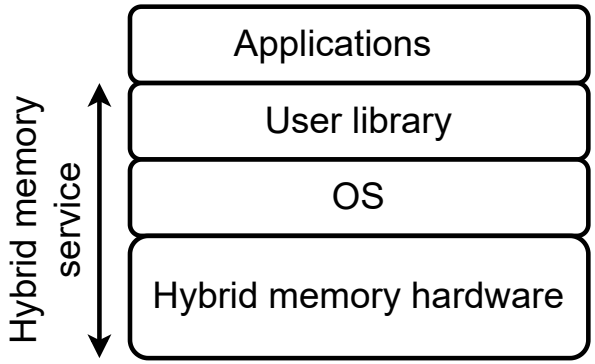


HSCC: Hardware/software cooperative caching for hybrid DRAM/NVM memory architectures

SSP: Eliminating redundant writes in failure-atomic NVRAMs via shadow sub-paging

Hybrid Memory System

- Hybrid memory service may span multiple layers.
- HSCC $\rightarrow$ page migration (OS + hardware)
- SSP $\rightarrow$ data persistence (OS + hardware)
- SoftWrAP $\rightarrow$ data persistence (user library)



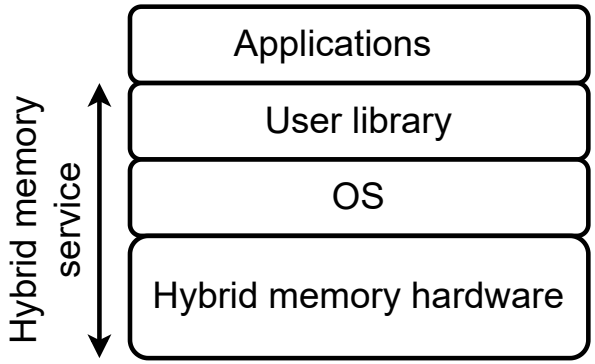
HSCC: Hardware/software cooperative caching for hybrid DRAM/NVM memory architectures

SSP: Eliminating redundant writes in failure-atomic NVRAMs via shadow sub-paging

Softwrap: A lightweight framework for transactional support of storage class memory

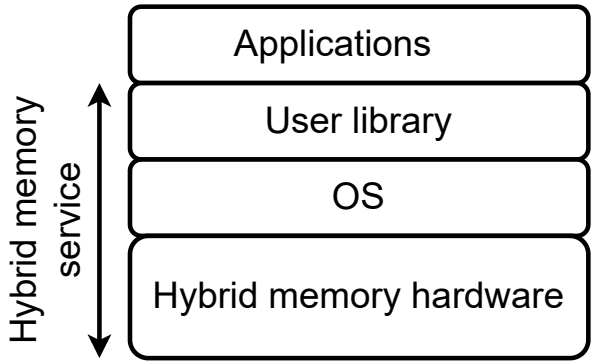
Hybrid Memory System

- Exploration of ideas in hybrid memory systems require a framework.



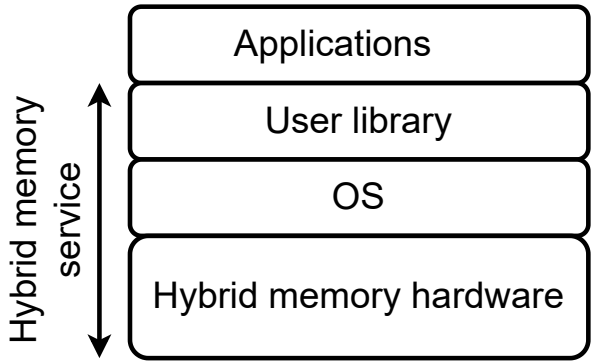
Hybrid Memory System

- Exploration of ideas in hybrid memory systems require a framework.
- Framework $\rightarrow$ cater to ideas at multiple layers.



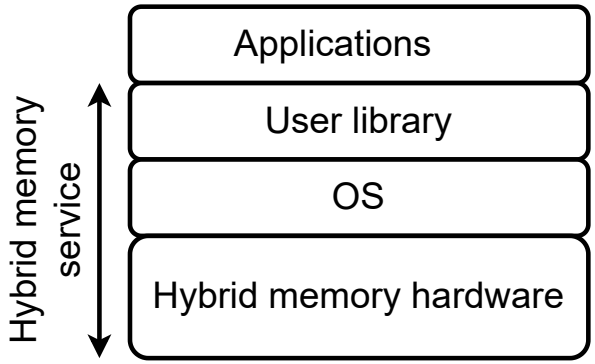
Hybrid Memory System

- Exploration of ideas in hybrid memory systems require a framework.
- Framework $\rightarrow$ cater to ideas at multiple layers.
- Framework $\rightarrow$ allow quick prototyping of ideas.



Hybrid Memory System

- Exploration of ideas in hybrid memory systems require a framework.
- Framework $\rightarrow$ cater to ideas at multiple layers.
- Framework $\rightarrow$ allow quick prototyping of ideas.
- Framework $\rightarrow$ hosts lightweight OS.



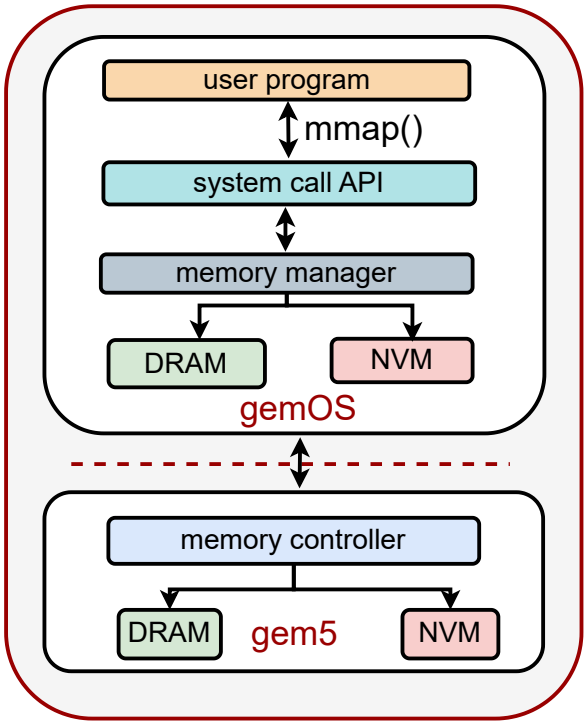
Kindle: Hybrid Memory Framework

Kindle: Hybrid Memory Framework

- Kindle is a lightweight full-system simulation framework for hybrid memory systems.

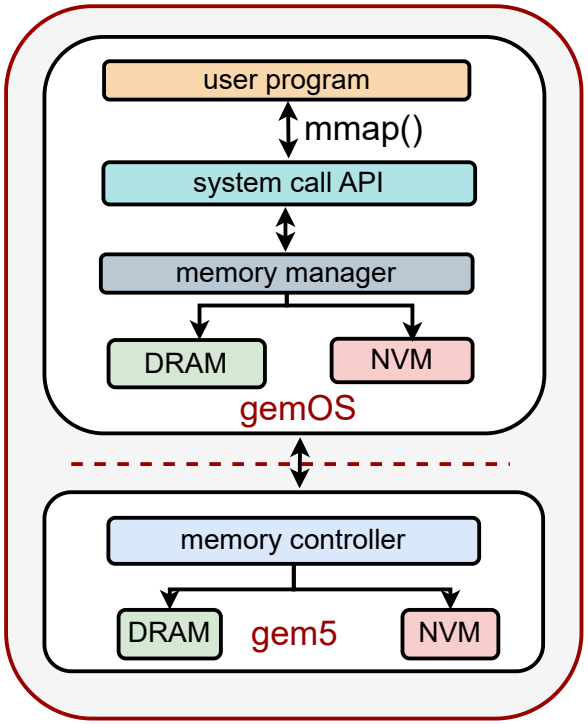
Kindle: Hybrid Memory Framework

- Kindle uses *gemOS* as OS component in framework.



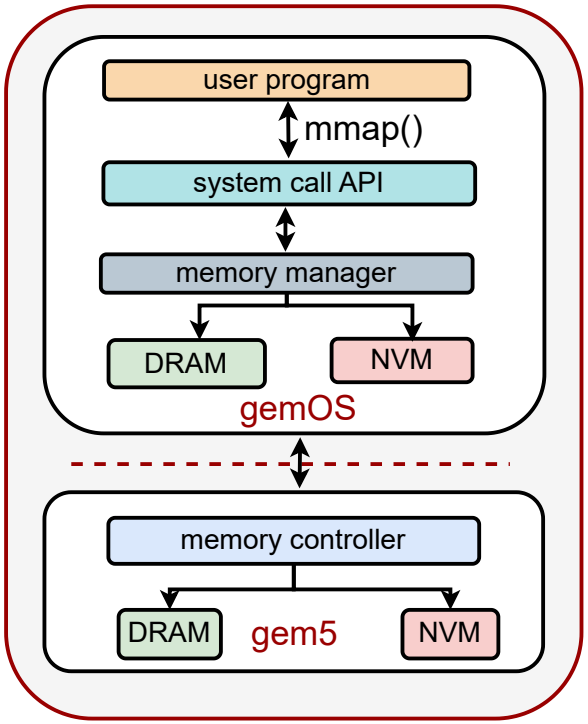
Kindle: Hybrid Memory Framework

- Kindle uses *gemOS* as OS component in framework.
- *gemOS* is lightweight, allows easy integration of hybrid memory support in OS.



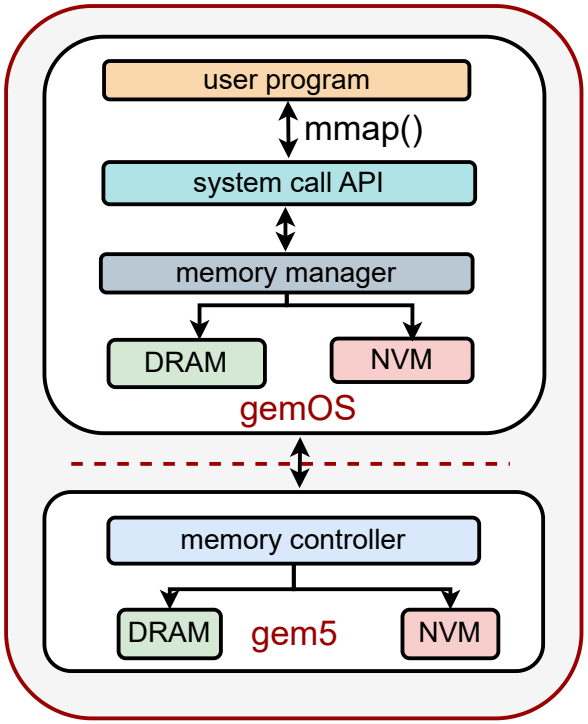
Kindle: Hybrid Memory Framework

- Kindle uses *gemOS* as OS component in framework.
- *gemOS* is lightweight, allows easy integration of hybrid memory support in OS.
- Application allocates memory from DRAM, NVM using memory allocation API.



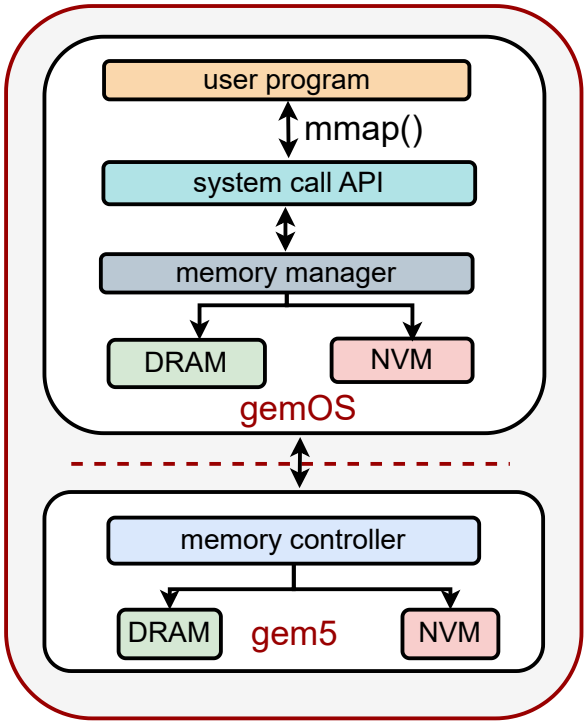
Kindle: Persistent Process Framework

- Kindle provides full process persistence in hybrid memory system.



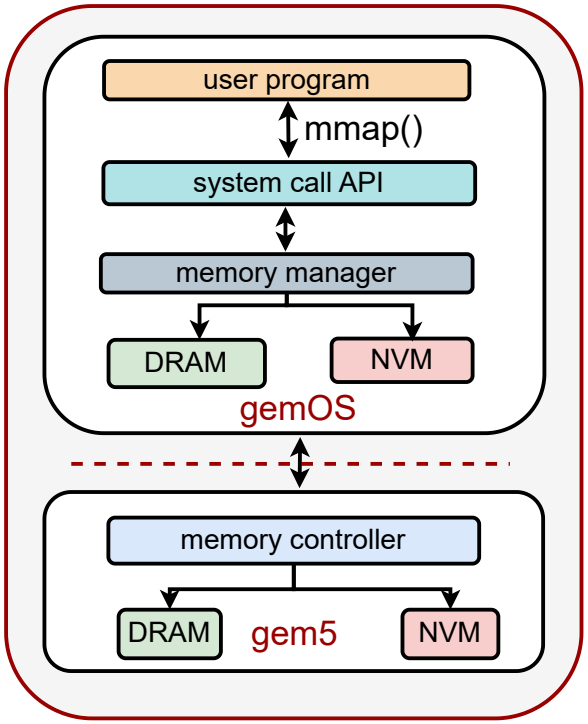
Kindle: Persistent Process Framework

- Kindle provides full process persistence in hybrid memory system.
- Kindle uses *gem5* as hardware component in framework.



Kindle: Persistent Process Framework

- Kindle provides full process persistence in hybrid memory system.
- Kindle uses *gem5* as hardware component in framework.
- Enables prototyping solutions crossing OS-Architecture boundaries.

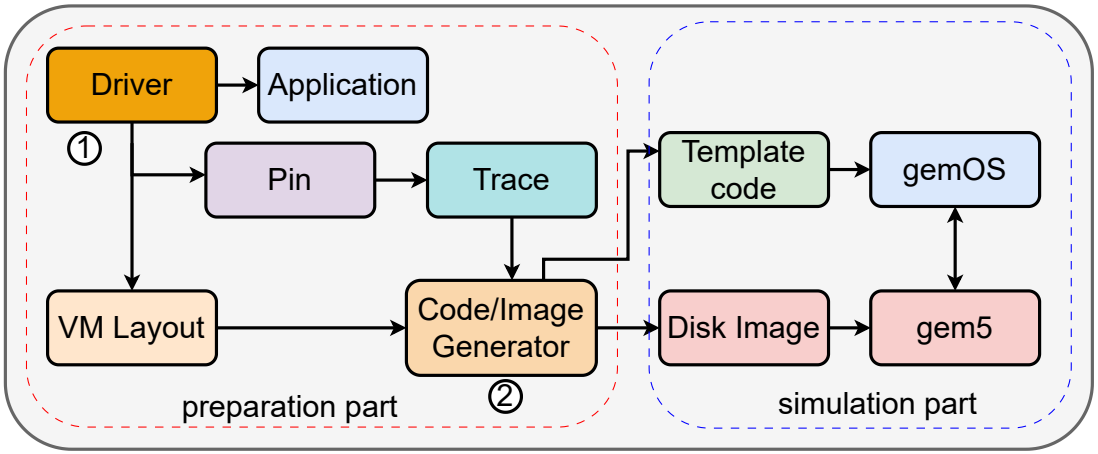


Kindle: Persistent Process Framework

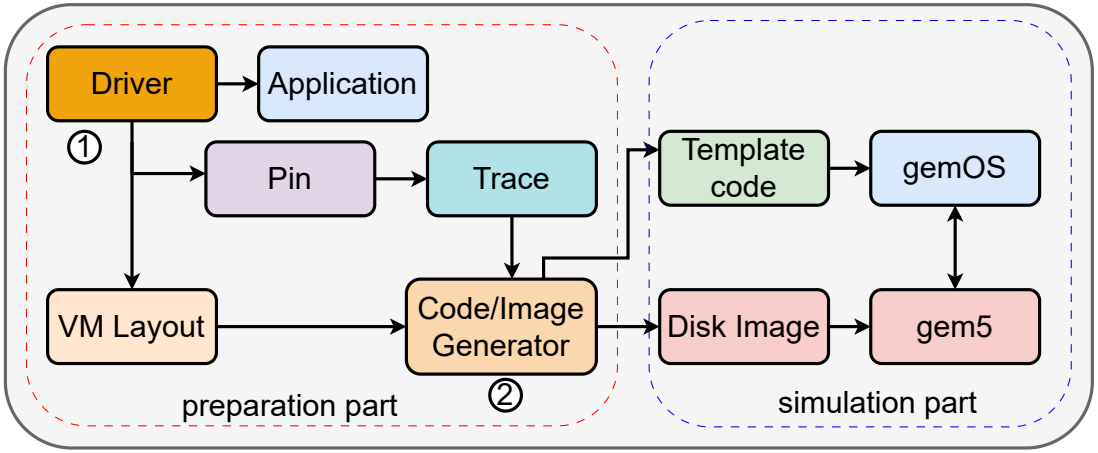
- Supports end-to-end study of real-world applications.

Kindle: Persistent Process Framework

- Supports end-to-end study of real-world applications.
- Kindle consists for two parts.

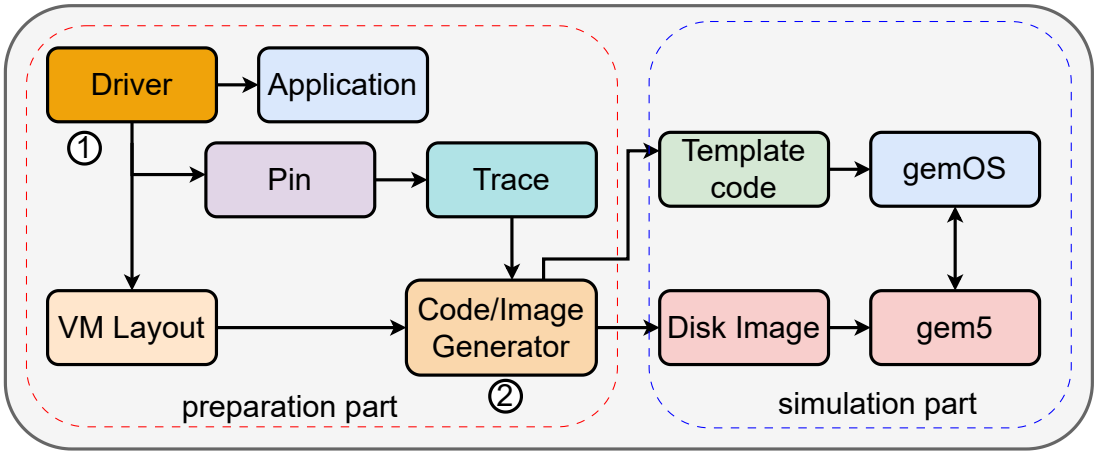


Kindle: Persistent Process Framework



- Preparation part → generate artifacts for simulation.

Kindle: Persistent Process Framework



- Preparation part → generate artifacts for simulation.
- Simulation part → run application with persistence.

Kindle: Page Table Consistency

Kindle: Page Table Consistency

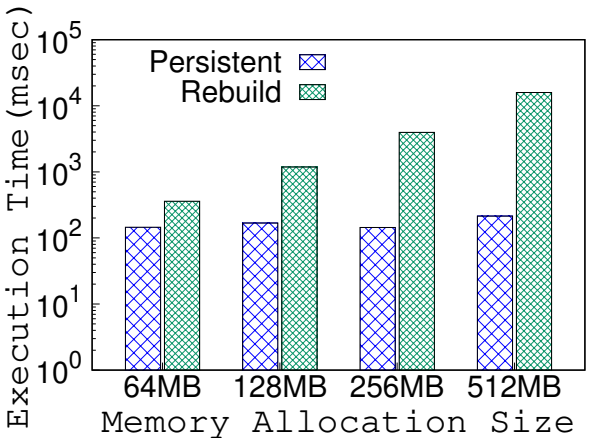
- Process persistence $\rightarrow$ consistent virtual address space management.

Kindle: Page Table Consistency

- Process persistence $\rightarrow$ consistent virtual address space management.
- Page table in NVM $\rightarrow$ Wrap modifications in failure atomicity (persistent scheme).
- Page table in DRAM $\rightarrow$ Checkpoint state to NVM periodically (rebuild scheme).

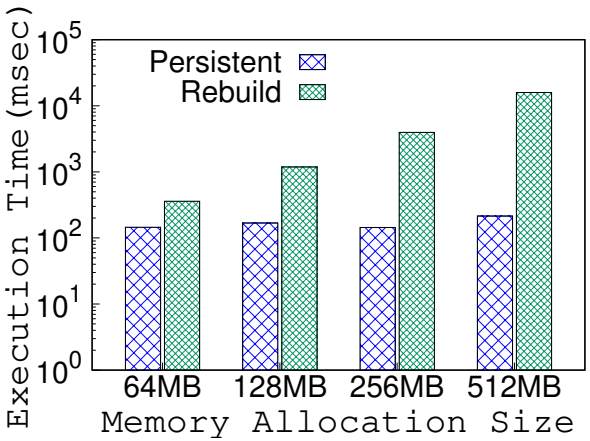
Kindle: Page Table Consistency

- Micro-benchmark sequentially access all pages in the allocated space.
- End-to-end execution time for consistently maintaining execution context.



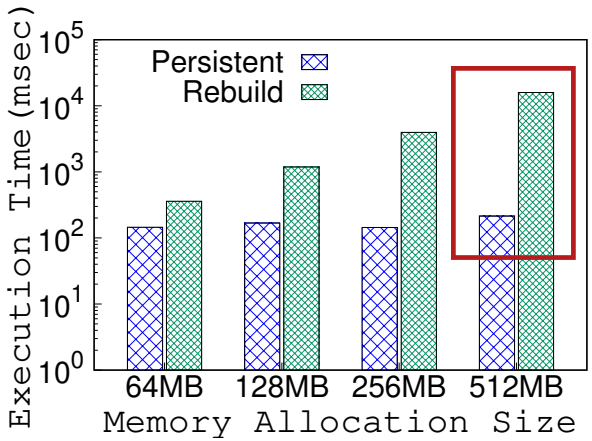
Kindle: Page Table Consistency

- Micro-benchmark sequentially access all pages in the allocated space.
- End-to-end execution time for consistently maintaining execution context.
- Rebuild scheme $\rightarrow$ higher execution time for all allocation sizes.



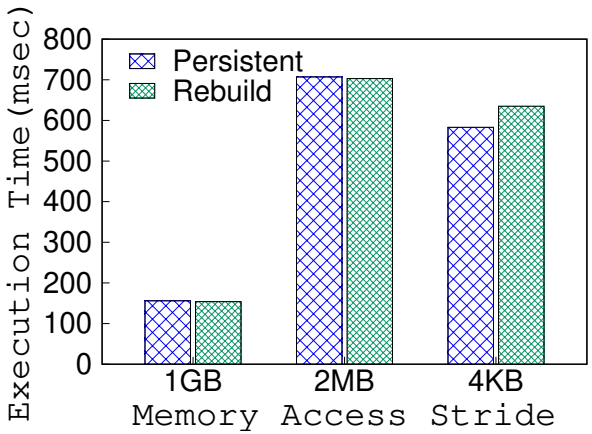
Kindle: Page Table Consistency

- Micro-benchmark sequentially access all pages in the allocated space.
- End-to-end execution time for consistently maintaining execution context.
- Rebuild scheme $\rightarrow$ higher execution time for all allocation sizes.
- Overhead $\rightarrow$ $\sim$74.2$\times$ (512MB) w.r.t Persistent.



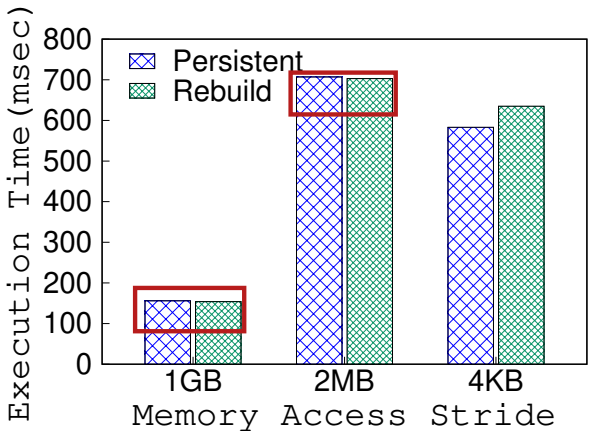
Kindle: Page Table Consistency

- Micro-benchmark stride access with predefined gap, e.g. 1GB, to map different page table levels.
- End-to-end execution time for consistently maintaining execution context.



Kindle: Page Table Consistency

- Micro-benchmark stride access with predefined gap, e.g. 1GB, to map different page table levels.
- End-to-end execution time for consistently maintaining execution context.
- Persistent $\rightarrow$ slightly more execution time compared to Rebuild for 1GB, 2MB.



Kindle: Page Table Consistency

Alloc/Free Size	Persistent (msec)	Rebuild (msec)
64MB	325	19377
128MB	389	23438
256MB	517	29376

- Micro-benchmark $\rightarrow$ sequence of fixed size `mmap` and `munmap` operations.
- End-to-end execution time for consistently maintaining execution context.

Kindle: Page Table Consistency

Alloc/Free Size	Persistent (msec)	Rebuild (msec)
64MB	325	19377
128MB	389	23438
256MB	517	29376

- Micro-benchmark $\rightarrow$ sequence of fixed size `mmap` and `munmap` operations.
- End-to-end execution time for consistently maintaining execution context.
- Persistent scheme overhead $\rightarrow$ $\sim$1.6$\times$ from 64MB to 256MB.

Kindle: Page Table Consistency

Alloc/Free Size	Persistent (msec)	Rebuild (msec)
64MB	325	19377
128MB	389	23438
256MB	517	29376

- Micro-benchmark $\rightarrow$ sequence of fixed size `mmap` and `munmap` operations.
- End-to-end execution time for consistently maintaining execution context.
- Persistent scheme overhead $\rightarrow \sim 1.6\times$ from 64MB to 256MB.
- Rebuild scheme overhead $\rightarrow \sim 1.5\times$ from 64MB to 256MB.

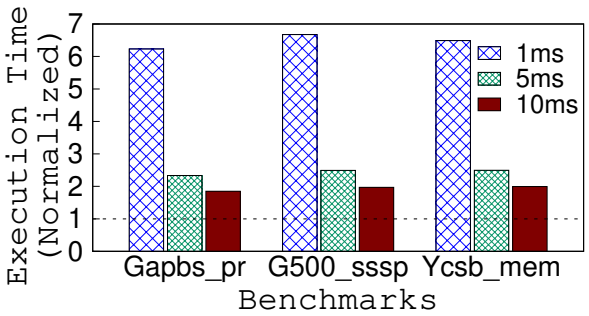
Existing work re-evaluation using Kindle

SSP using Kindle

- SSP provides NVM memory consistency at cache line granularity.

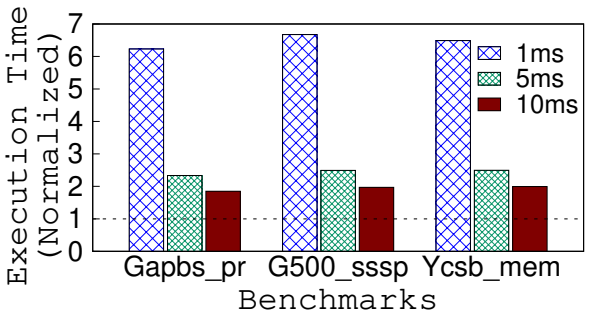
SSP using Kindle

- SSP provides NVM memory consistency at cache line granularity.
- Used periodic consistency interval of 1, 5 and 10 msec.



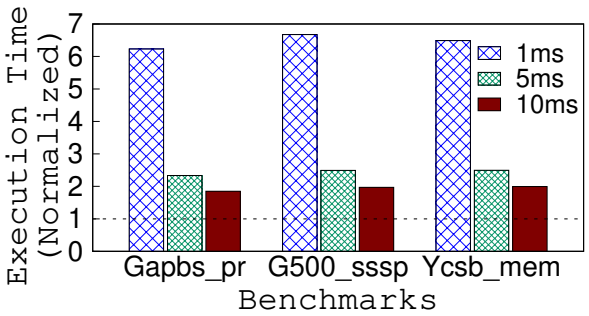
SSP using Kindle

- SSP provides NVM memory consistency at cache line granularity.
- Used periodic consistency interval of 1, 5 and 10 msec.
- Page consolidation thread is fixed to 1 msec.



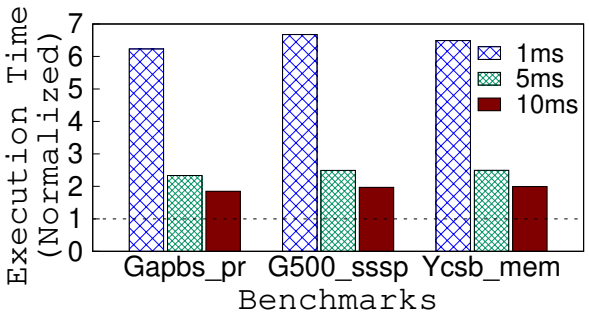
SSP using Kindle

- SSP provides NVM memory consistency at cache line granularity.
- Used periodic consistency interval of 1, 5 and 10 msec.
- Page consolidation thread is fixed to 1 msec.
- Execution time with SSP normalized to time with no memory consistency.



SSP using Kindle

- SSP provides NVM memory consistency at cache line granularity.
- Used periodic consistency interval of 1, 5 and 10 msec.
- Page consolidation thread is fixed to 1 msec.
- Execution time with SSP normalized to time with no memory consistency.
  - Average $\sim 3\times$ reduction in overhead from 1 msec to 10 msec.



HSCC using Kindle

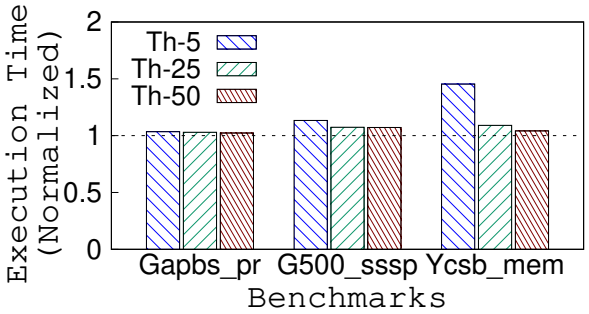
- HSCC uses DRAM as cache managed by OS.

HSCC using Kindle

- HSCC uses DRAM as cache managed by OS.
- HSCC tracks access counts to NVM pages to select for migration.

HSCC using Kindle

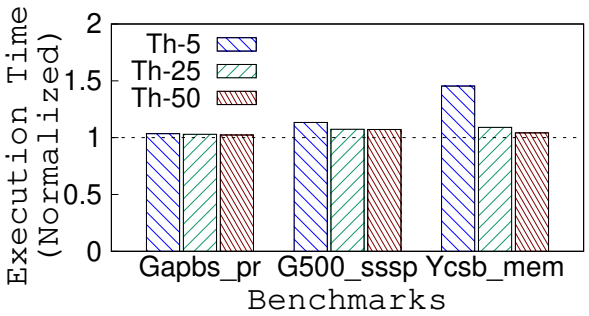
- HSCC uses DRAM as cache managed by OS.
- HSCC tracks access counts to NVM pages to select for migration.



- Execution time with OS migration activities normalized to time without OS activities.

HSCC using Kindle

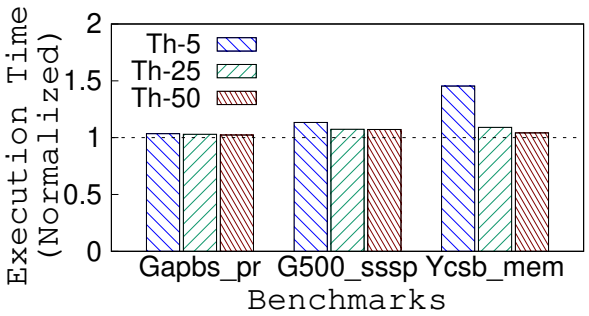
- HSCC uses DRAM as cache managed by OS.
- HSCC tracks access counts to NVM pages to select for migration.



- Execution time with OS migration activities normalized to time without OS activities.
- Migration interval is 31.25 msec and fetch thresholds are 5, 25 and 50

HSCC using Kindle

- HSCC uses DRAM as cache managed by OS.
- HSCC tracks access counts to NVM pages to select for migration.



- Execution time with OS migration activities normalized to time without OS activities.
- Migration interval is 31.25 msec and fetch thresholds are 5, 25 and 50
- Gapbs_pr shows the minimum overhead.

HSCC using Kindle

Benchmark	Th-5	Th-25	Th-50
Gapbs_pr	354	273	132
G500_sssp	4489	1475	1346
Ycsb_mem	23093	1661	221

- Table shows number of pages migrated for different thresholds.

HSCC using Kindle

Benchmark	Th-5	Th-25	Th-50
Gapbs_pr	354	273	132
G500_sssp	4489	1475	1346
Ycsb_mem	23093	1661	221

- Table shows number of pages migrated for different thresholds.
- Number of pages migrated reduces with increase in fetch threshold.

HSCC using Kindle

Benchmark	Th-5	Th-25	Th-50
Gapbs_pr	354	273	132
G500_sssp	4489	1475	1346
Ycsb_mem	23093	1661	221

- Table shows number of pages migrated for different thresholds.
- Number of pages migrated reduces with increase in fetch threshold.
- Ycsb_mem $\rightarrow \sim 13\times$ reduction in number of pages migrated for Th-25 compared to Th-5.

HSCC using Kindle

Benchmark	Fetch Threshold	Page Selection (%)	Page Copy (%)
Gapbs_pr	Th-5	1.74	98.26
	Th-25	1.92	98.08
	Th-50	2.06	97.94

- Migrating a page to DRAM consists of *page selection* and *page copy*

HSCC using Kindle

Benchmark	Fetch Threshold	Page Selection (%)	Page Copy (%)
Gapbs_pr	Th-5	1.74	98.26
	Th-25	1.92	98.08
	Th-50	2.06	97.94

- Migrating a page to DRAM consists of *page selection* and *page copy*
- Table shows percentage of time spent as part of OS migration activity.

HSCC using Kindle

Benchmark	Fetch Threshold	Page Selection (%)	Page Copy (%)
Gapbs_pr	Th-5	1.74	98.26
	Th-25	1.92	98.08
	Th-50	2.06	97.94

- Migrating a page to DRAM consists of *page selection* and *page copy*
- Table shows percentage of time spent as part of OS migration activity.
- Gapbs_pr $\rightarrow$ Page selection time is less than $\sim$2% across all fetch thresholds.

HSCC using Kindle

Benchmark	Fetch Threshold	Page Selection (%)	Page Copy (%)
Gapbs_pr	Th-5	1.74	98.26
	Th-25	1.92	98.08
	Th-50	2.06	97.94

- Migrating a page to DRAM consists of *page selection* and *page copy*
- Table shows percentage of time spent as part of OS migration activity.
- Gapbs_pr $\rightarrow$ Page selection time is less than $\sim$2% across all fetch thresholds.
- Page copy occupies majority of execution time across all benchmarks.

Conclusion

- Kindle enables hybrid memory exploration crossing architecture-OS boundary.

Conclusion

- Kindle enables hybrid memory exploration crossing architecture-OS boundary.
- Kindle uses gemOS to circumvent the challenges of integrating NVM support in conventional OS for gem5.

Conclusion

- Kindle enables hybrid memory exploration crossing architecture-OS boundary.
- Kindle uses gemOS to circumvent the challenges of integrating NVM support in conventional OS for gem5.
- Kindle provides full process persistence.

Conclusion

- Kindle enables hybrid memory exploration crossing architecture-OS boundary.
- Kindle uses gemOS to circumvent the challenges of integrating NVM support in conventional OS for gem5.
- Kindle provides full process persistence.
- We compared two schemes to consistently maintain page table using Kindle.

Conclusion

- Kindle enables hybrid memory exploration crossing architecture-OS boundary.
- Kindle uses gemOS to circumvent the challenges of integrating NVM support in conventional OS for gem5.
- Kindle provides full process persistence.
- We compared two schemes to consistently maintain page table using Kindle.
- We showed utility of Kindle with two prototype implementations of state-of-the-art hybrid memory schemes, SSP and HSCC.

Questions?

Scan for Kindle