PURDUE UNIVERSITY

# ReplayCache: Enabling Caches for Energy Harvesting Systems

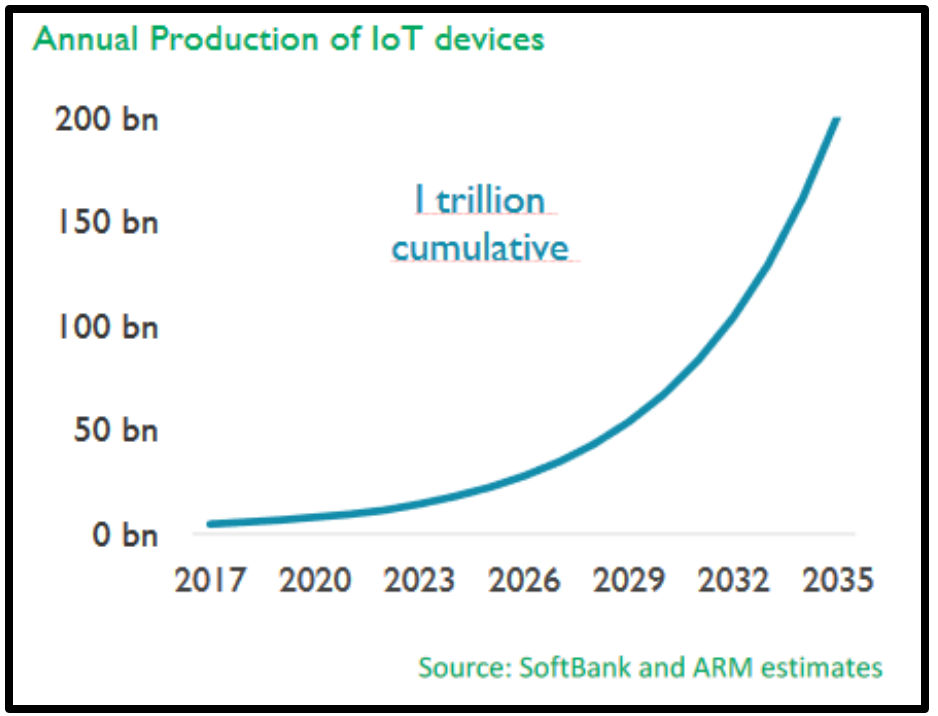
Jianping Zeng[1], Jongouk Choi[1], Xinwei Fu[2],
Ajay Paddayuru Shreepathi[3], Dongyoon Lee[3],
Changwoo Min[2], Changhee Jung[1]

[1]Purdue University
[2]Virginia Tech
[3]Stony Brook University

# Battery is Not the Way to Go!

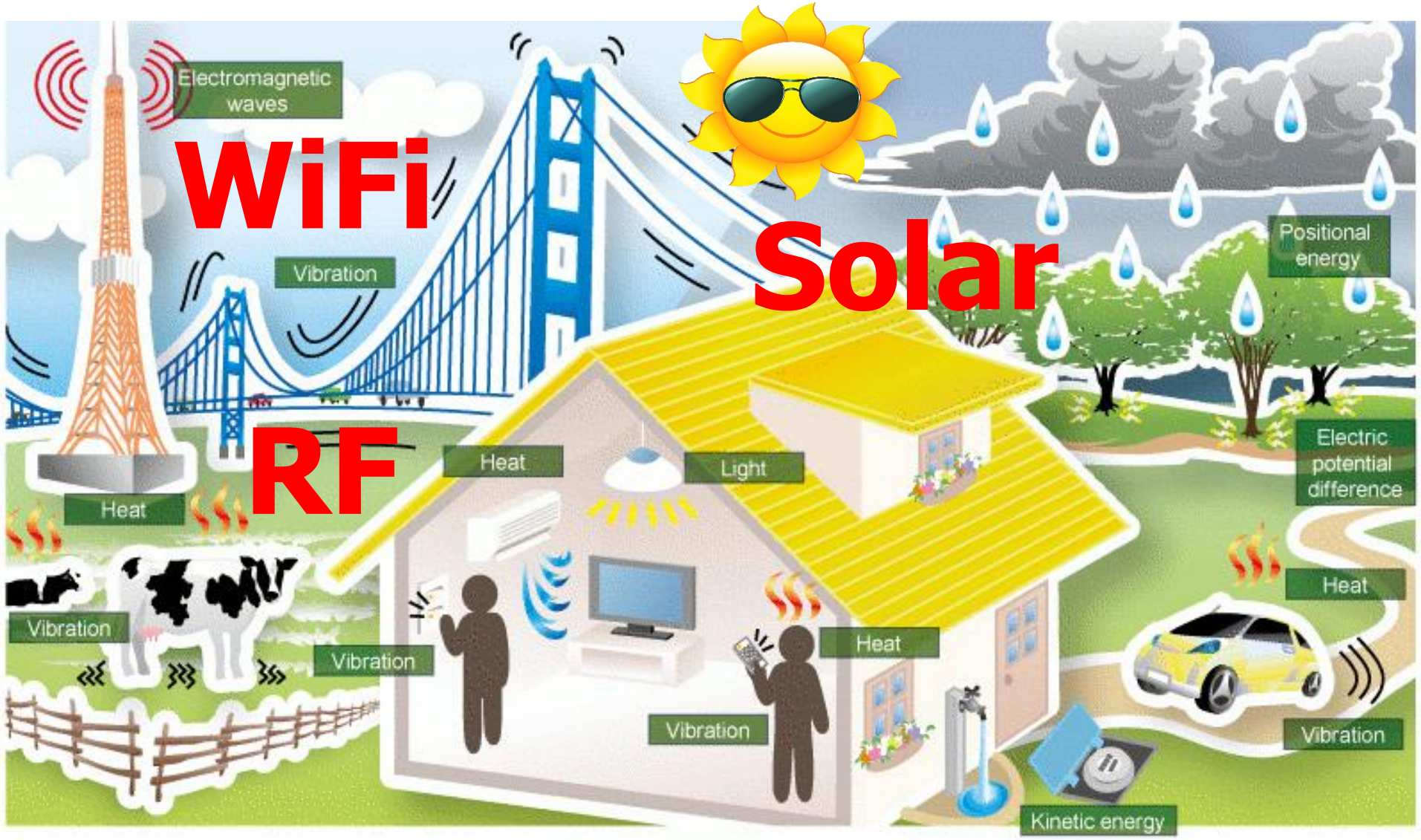
PURDUE UNIVERSITY

- Batteries are bulky
- They must be replaced



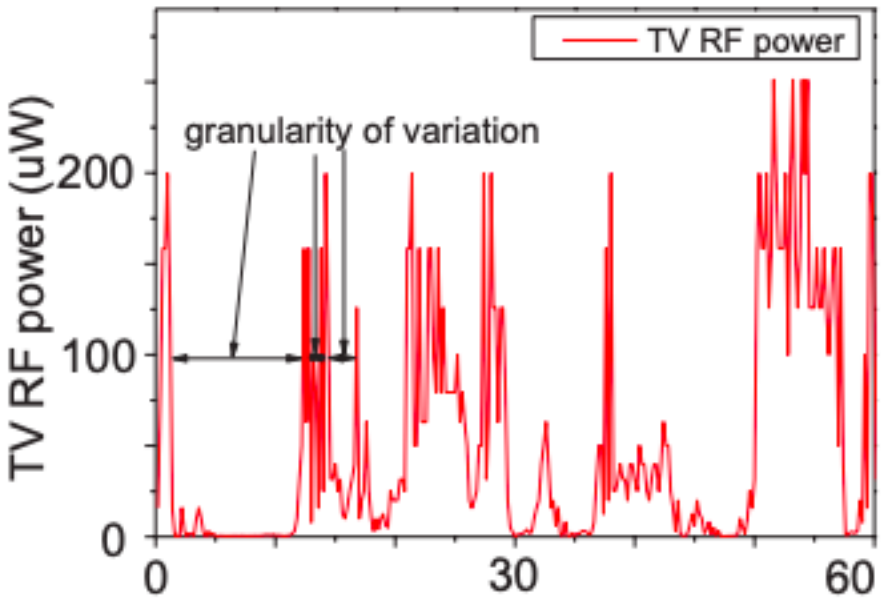
# Energy Harvesting Systems (EHS)

PURDUE UNIVERSITY

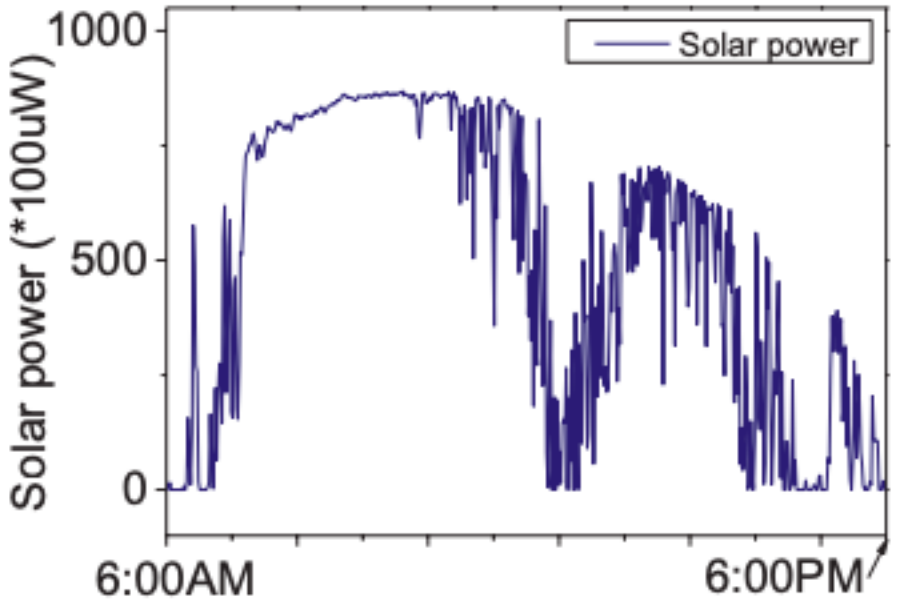


# Unreliability of Ambient Energy Sources

PURDUE UNIVERSITY




(a). Time (s) (Sample time: 0.33us)

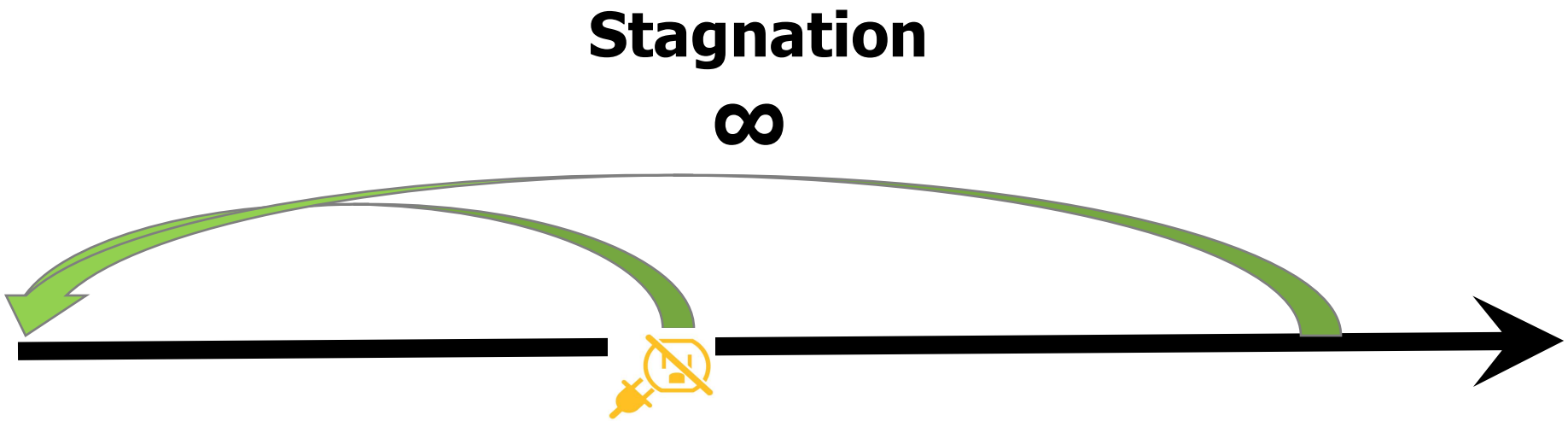



(b). Time (s) (Sample time: 0.33us)

[Ma,HPCA'2015]

# Problem of Frequently Data Loss

PURDUE UNIVERSITY

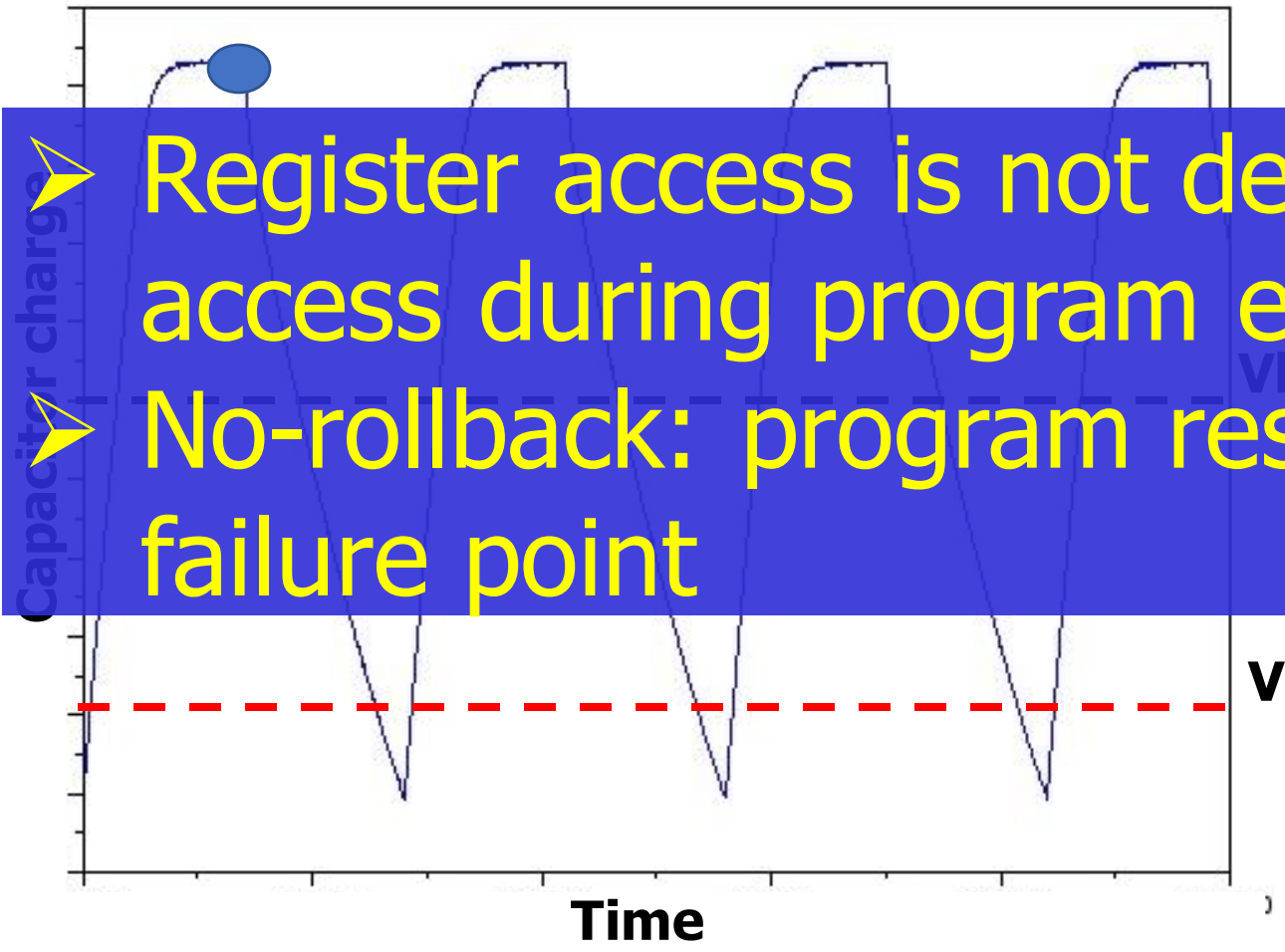



Timeline of program execution

: Power failure on which all volatile registers lose their data

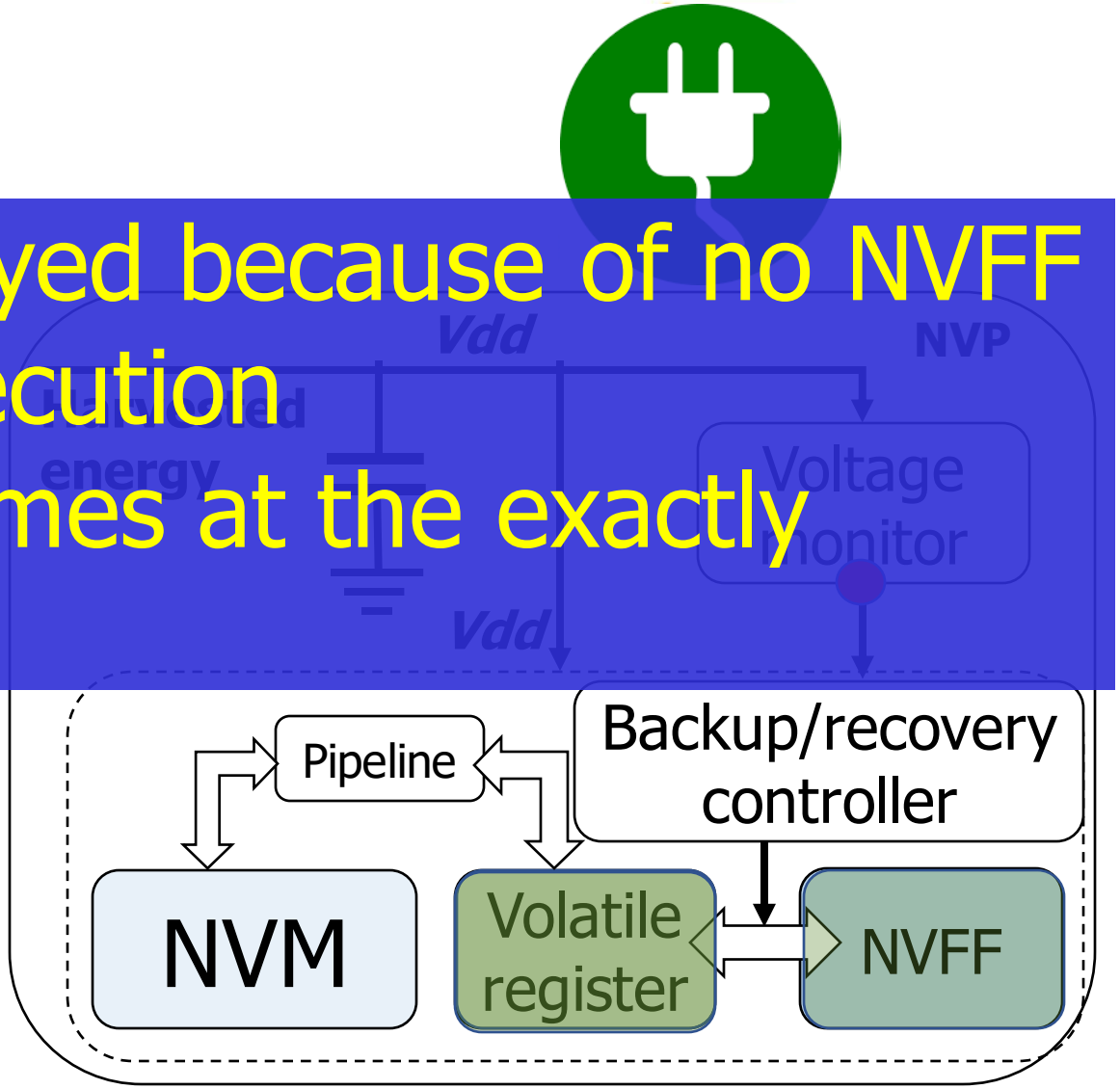
# NVP: Just-in-time Register Checkpointing

PURDUE UNIVERSITY




[Ma,HPCA'2015]

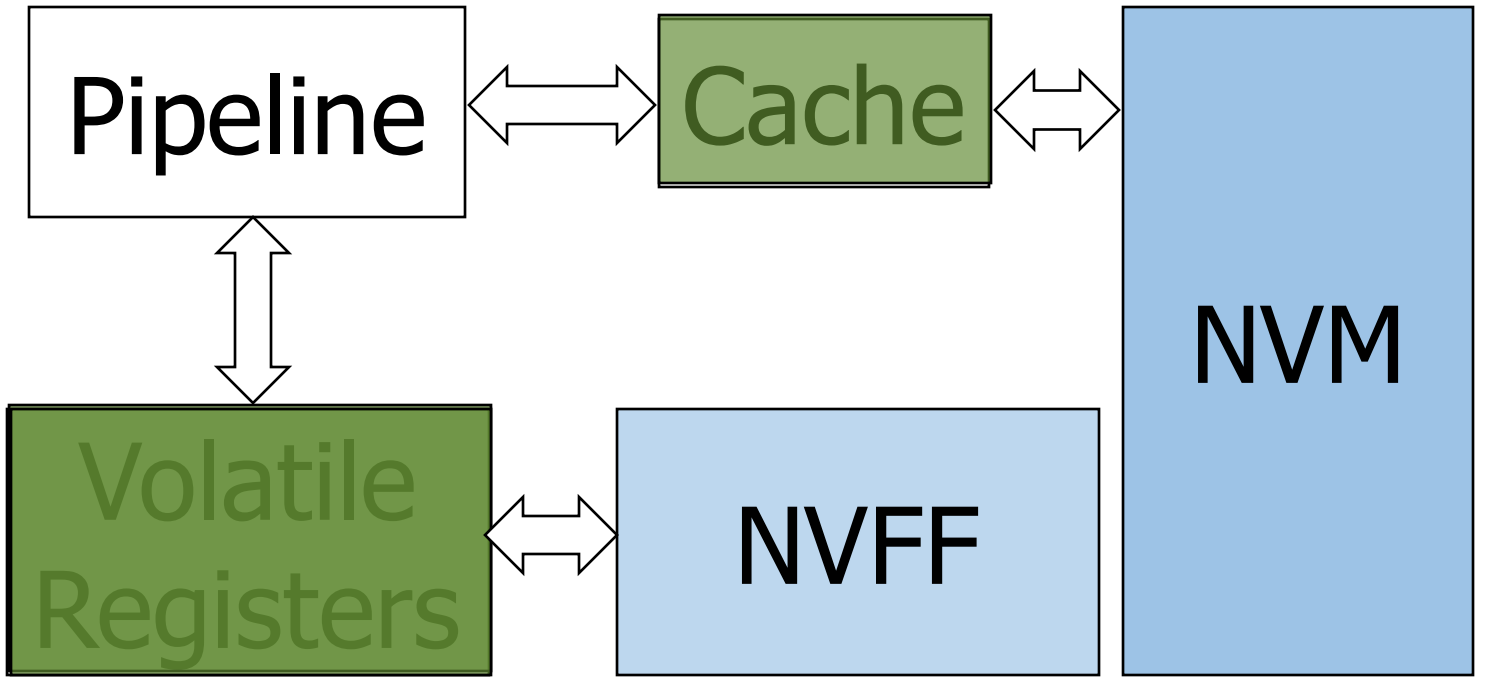
- Register access is not delayed because of no NVFF access during program execution
- No-rollback: program resumes at the exactly failure point



# NVP Has No Cache Due to Its Crash Consistency Issue

PURDUE UNIVERSITY

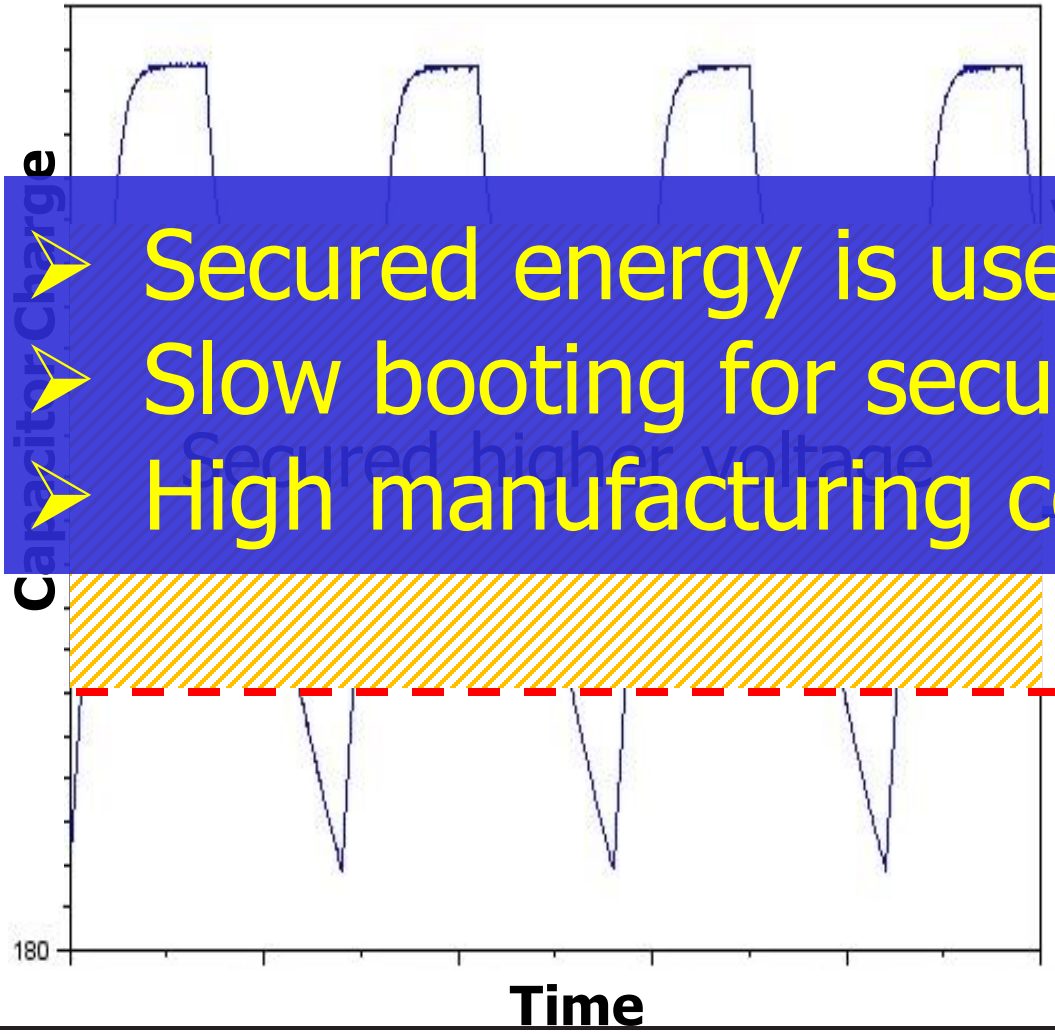
* We can't restart program from the exactly failure point as NVP does



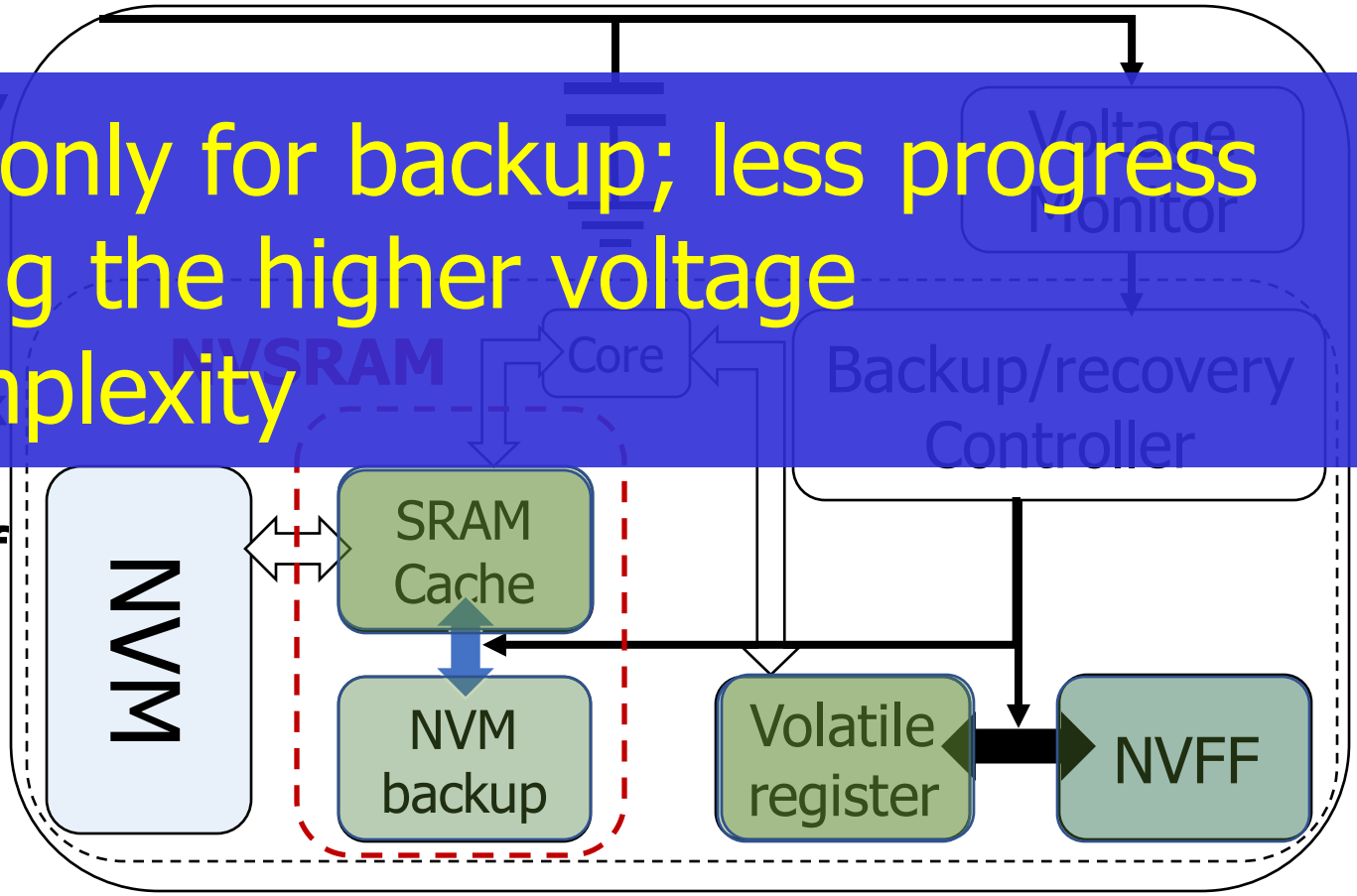

# NVSRAM: The State-of-the-Art and its Limitations

PURDUE UNIVERSITY

[Chui, VLSI'2010]

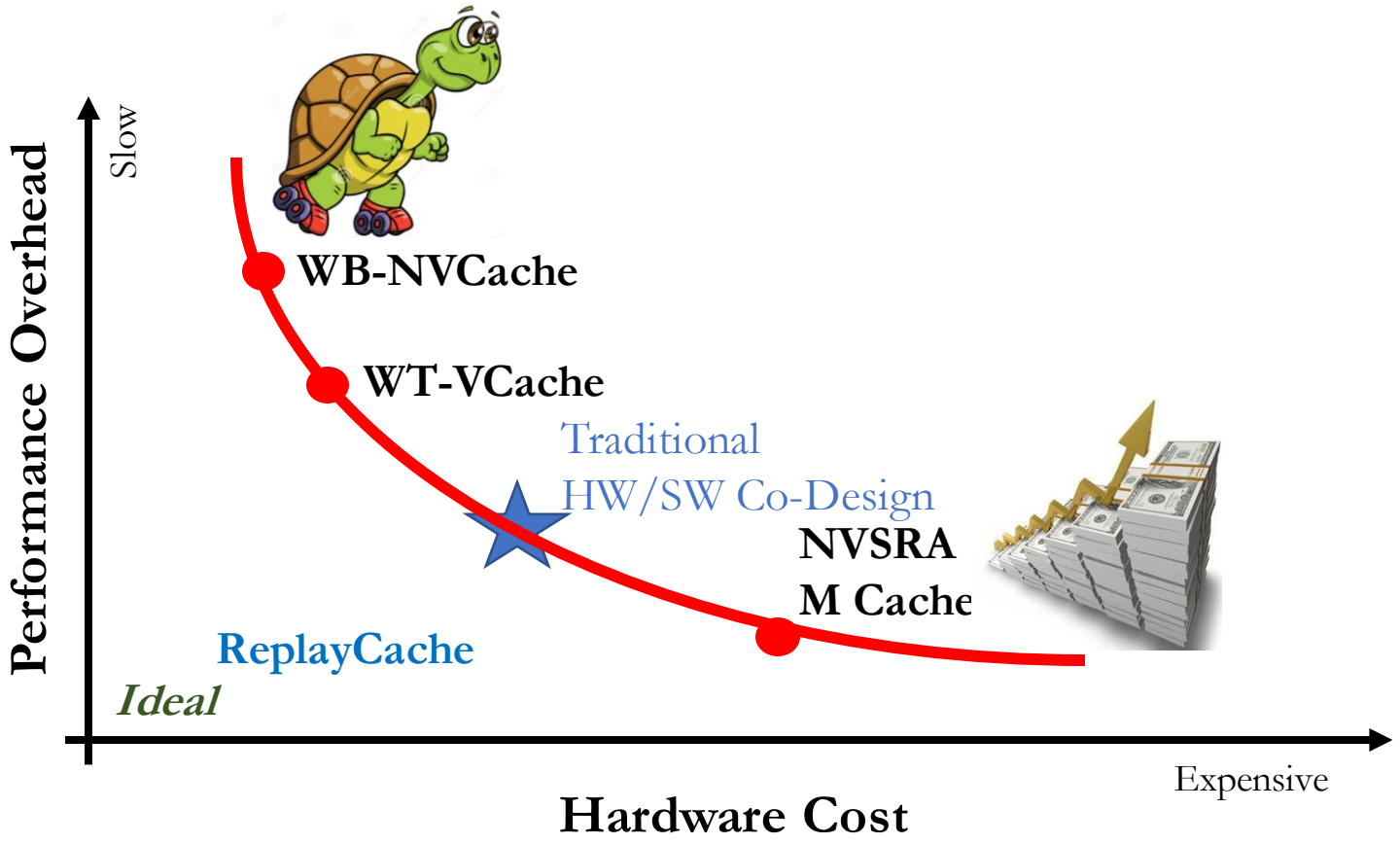



- Secured energy is used only for backup; less progress
- Slow booting for securing the higher voltage
- High manufacturing complexity



# ReplayCache: A Pure Software Solution to Enabling Performant Volatile Cache for EHS

PURDUE UNIVERSITY

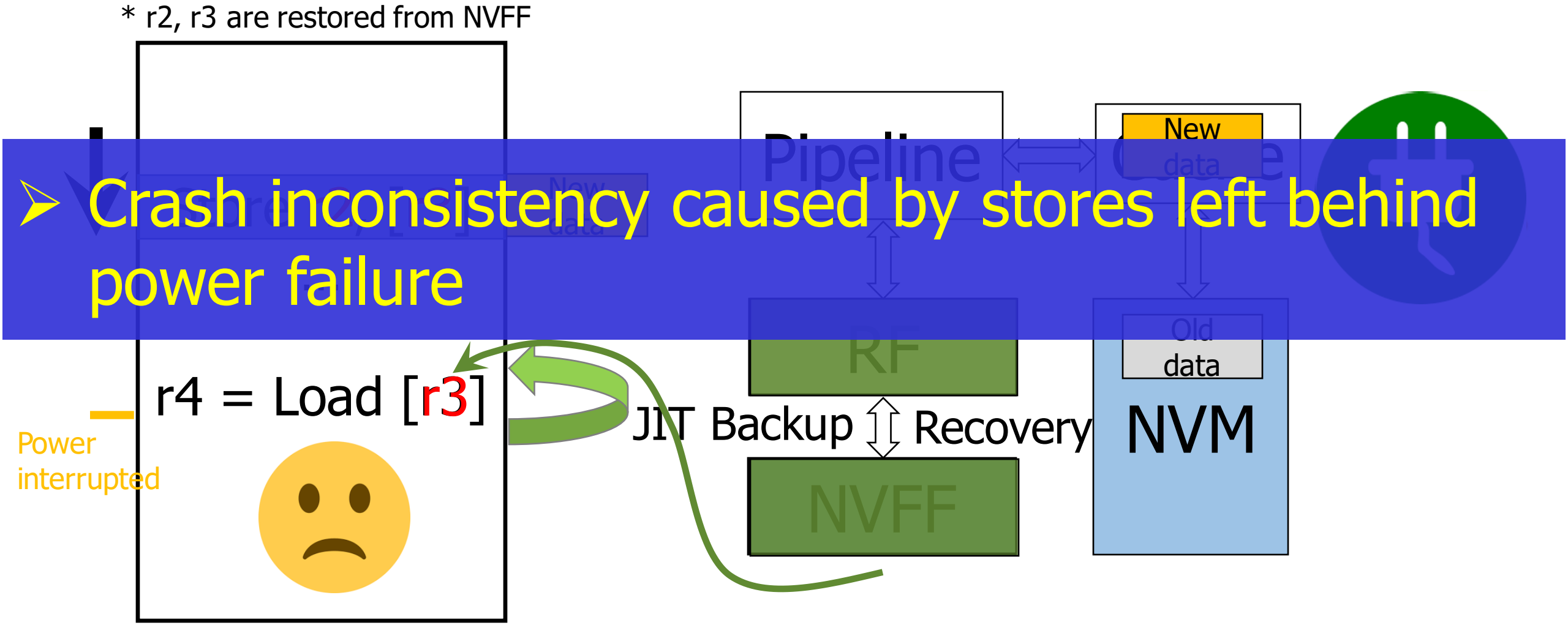



* Software undo/redo logging slow down 1.6-5x

# Reason for Crash Inconsistency of Volatile Cache

PURDUE UNIVERSITY

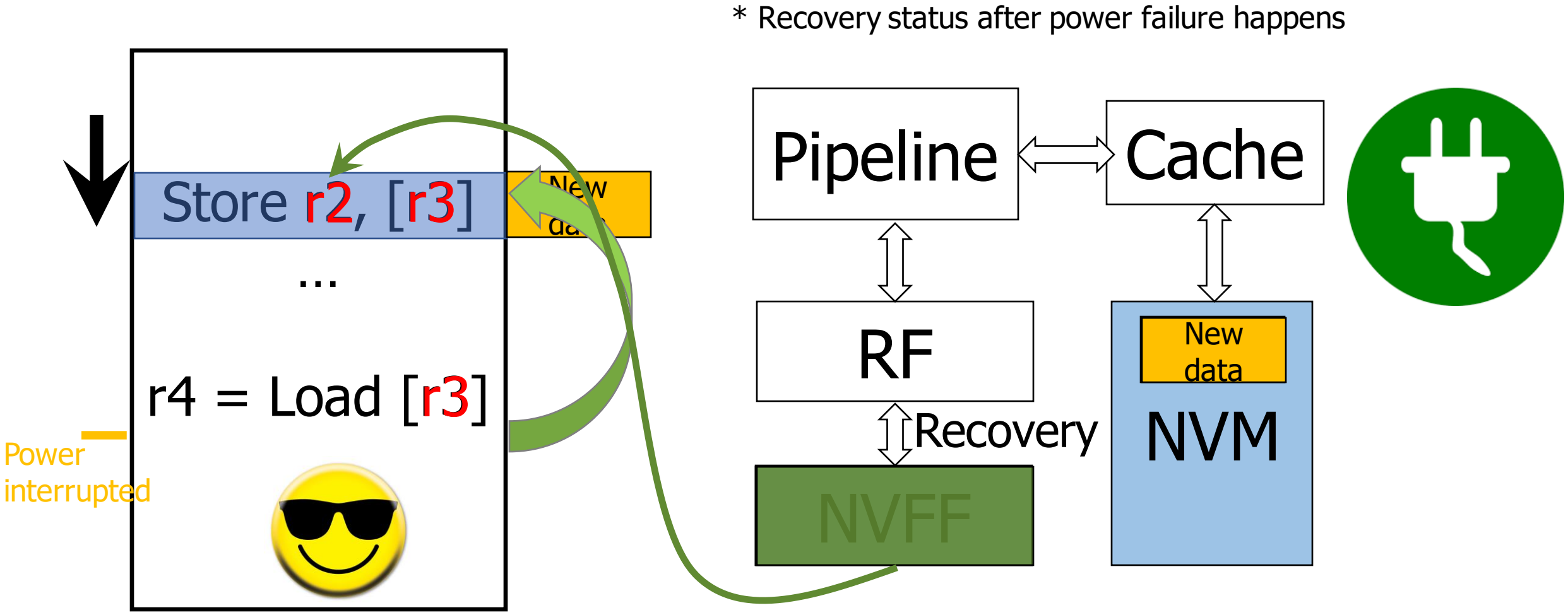
* r2, r3 are restored from NVFF




- Crash inconsistency caused by stores left behind power failure

# ReplayCache Solution: Replaying Unpersisted Stores

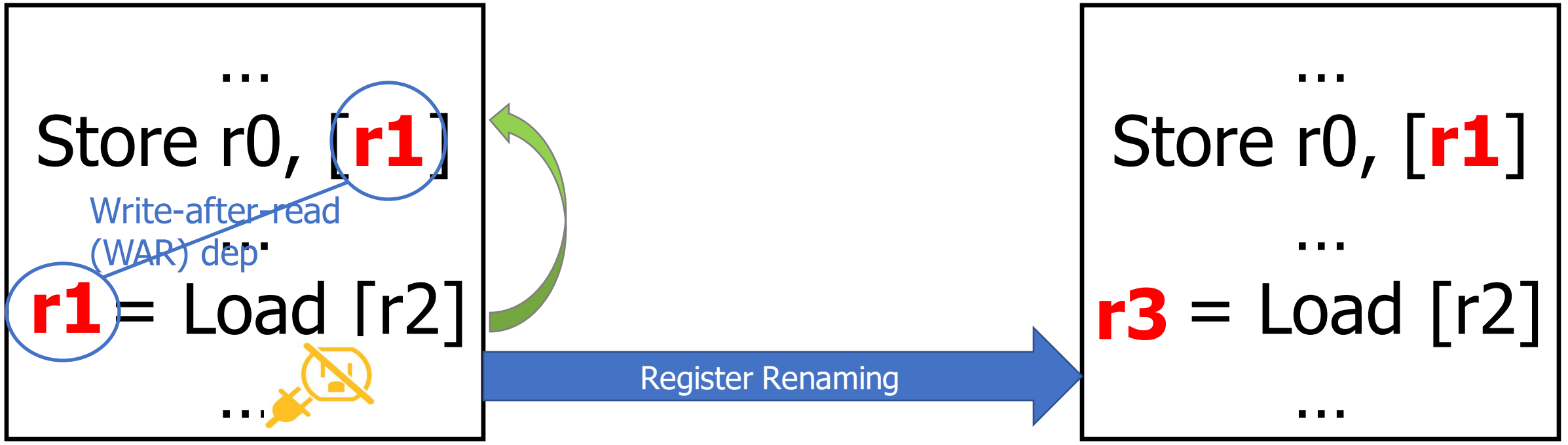
PURDUE UNIVERSITY



# Store Integrity: A Property to Ensure Correct Replaying

PURDUE UNIVERSITY

- Store integrity: register operands of stores must be not overwritten by following definitions



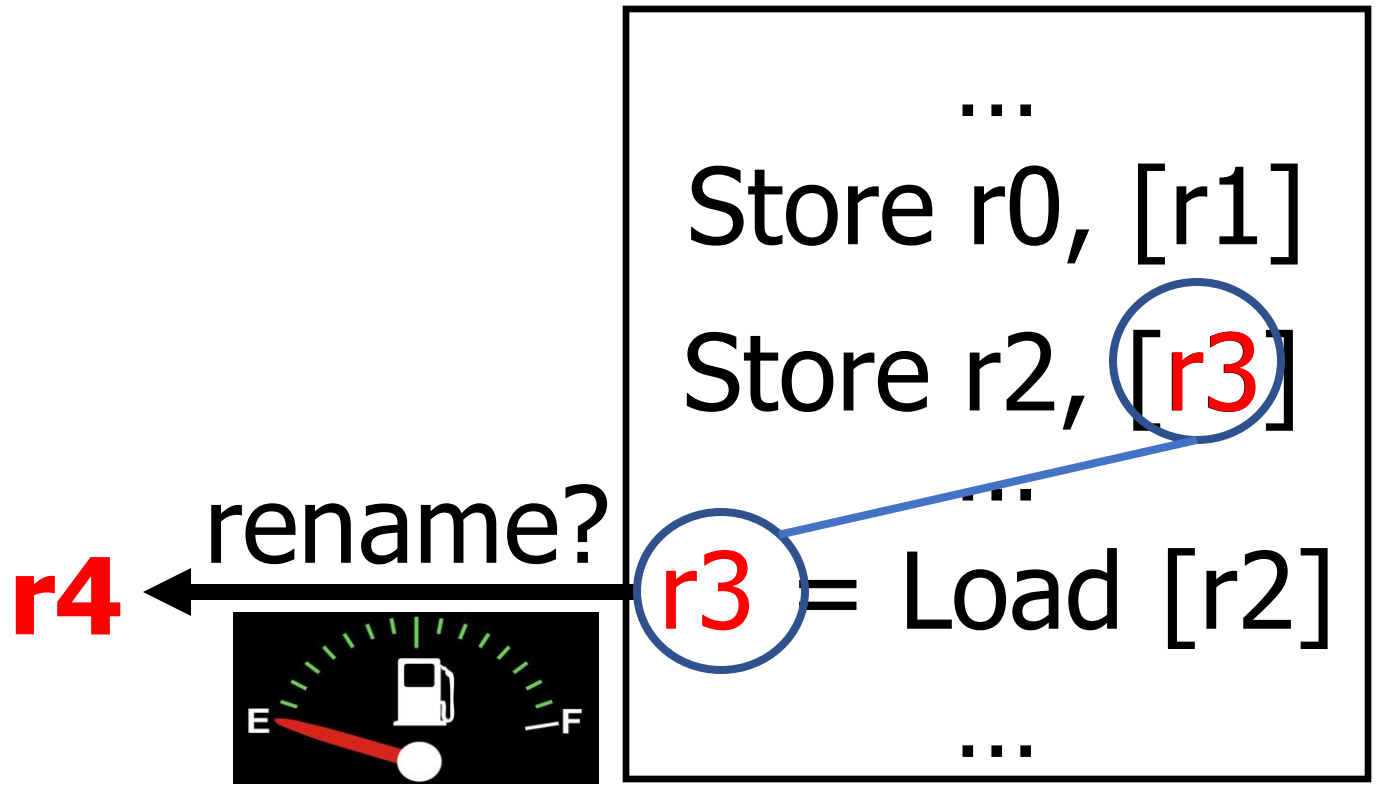

(a). Code of no store integrity

(b). Store integrity code by renaming

# Challenge of Guaranteeing Store Integrity

PURDUE UNIVERSITY

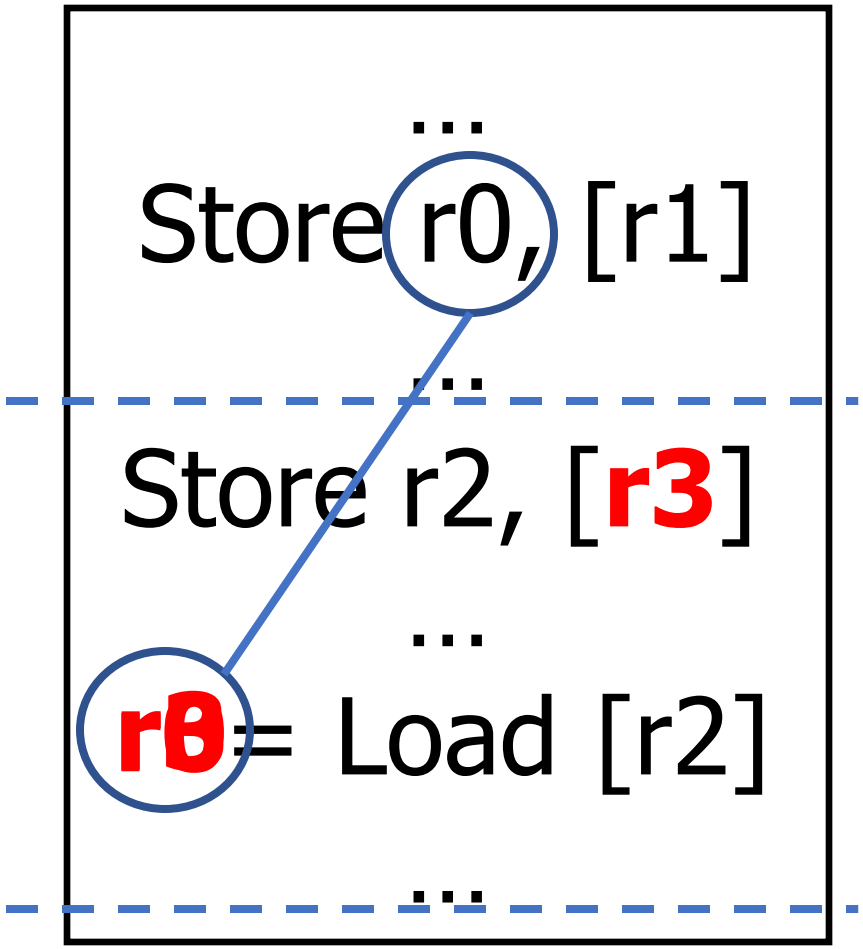
- Limited number of registers prevents store integrity




Assume only 4 registers

Write-after-read (WAR) dep

# Region-Level Store Integrity and Implications

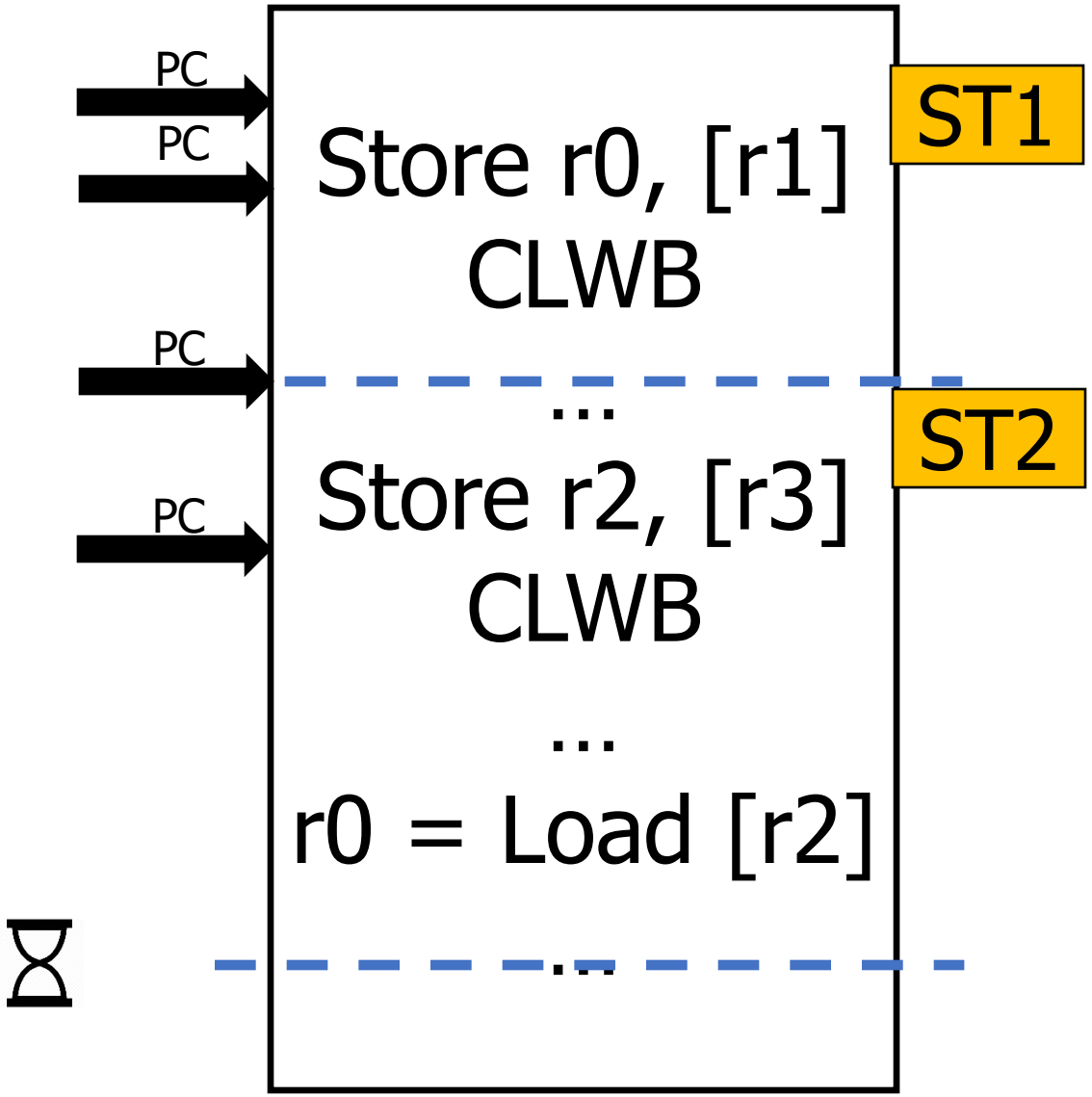



Region level register reuse

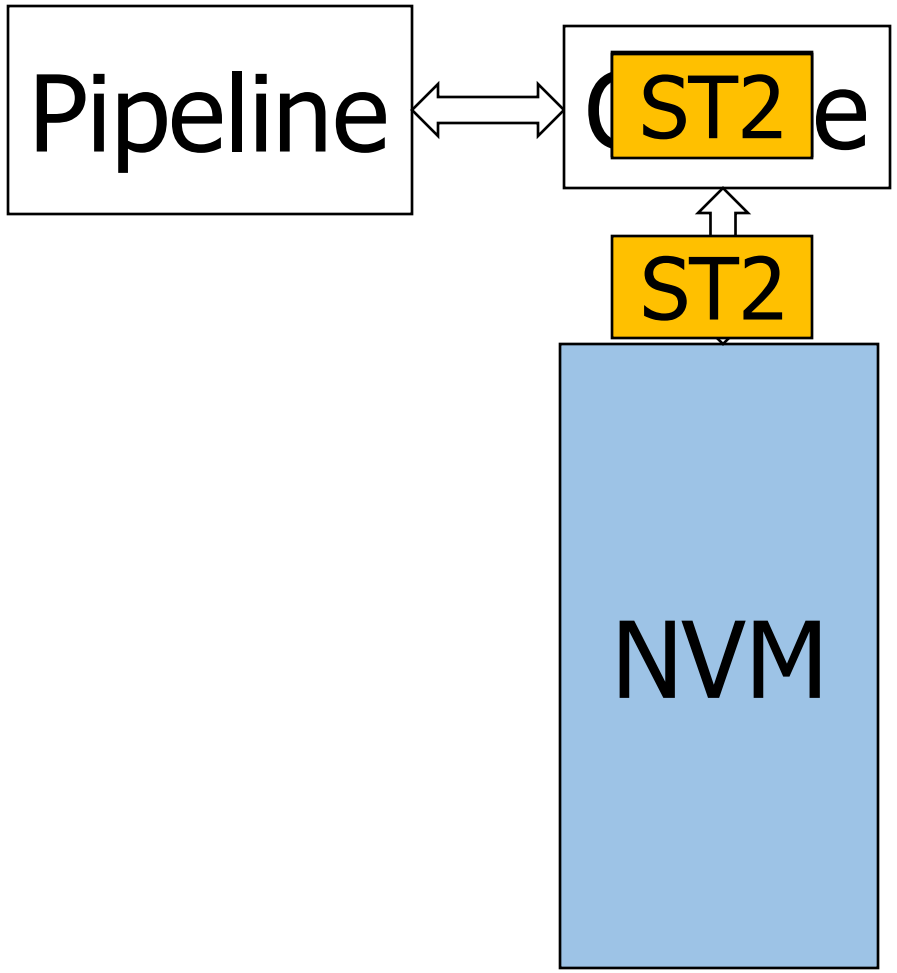
Region level store persistence

- Safe to overwrite store register across regions

# Region-Level Store Persistence




* CLWB: asynchronously write back store to memory.



# Recovery Protocol

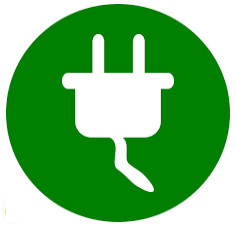
PURDUE UNIVERSITY

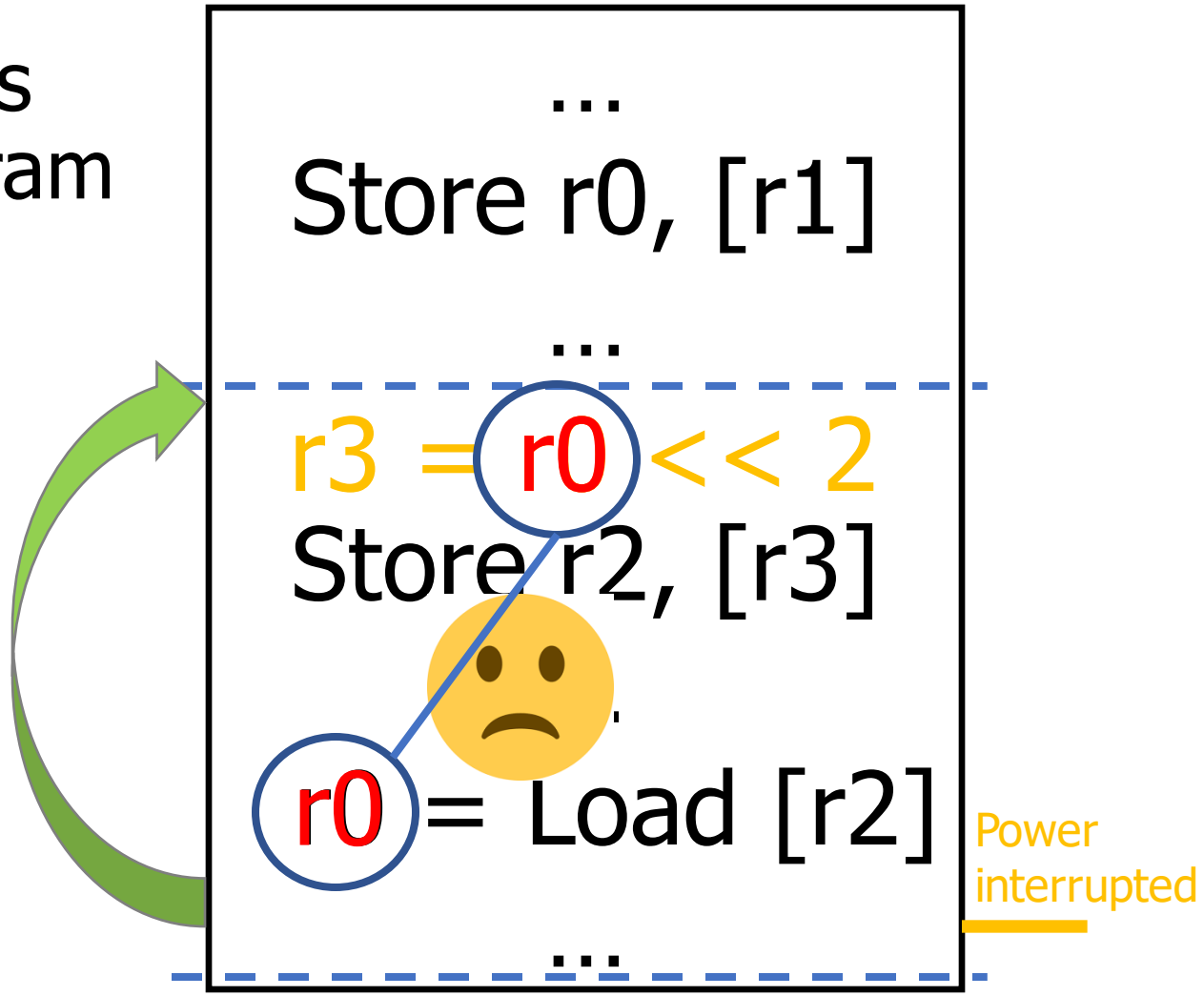
- Stores left behind failure are the root of cause
- Replaying them in the wake of failure with store integrity
- Due to limited registers, region-level store integrity is introduced, which in turn requires region-level store persistence to allow each region to use all registers.
- **Recovery protocol**

# Just Restart a Power-interrupted Region?

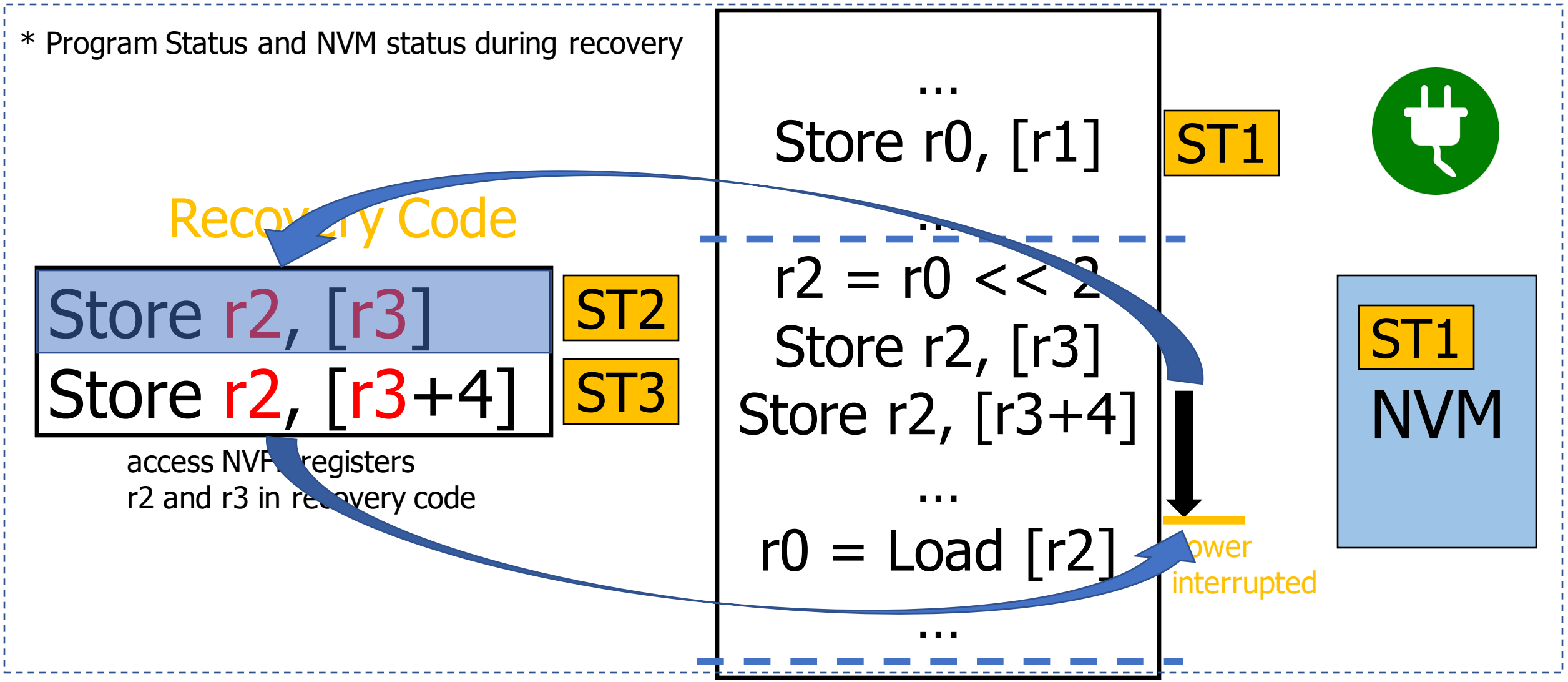
PURDUE UNIVERSITY

- Re-executing the region from its beginning fails to recover program status

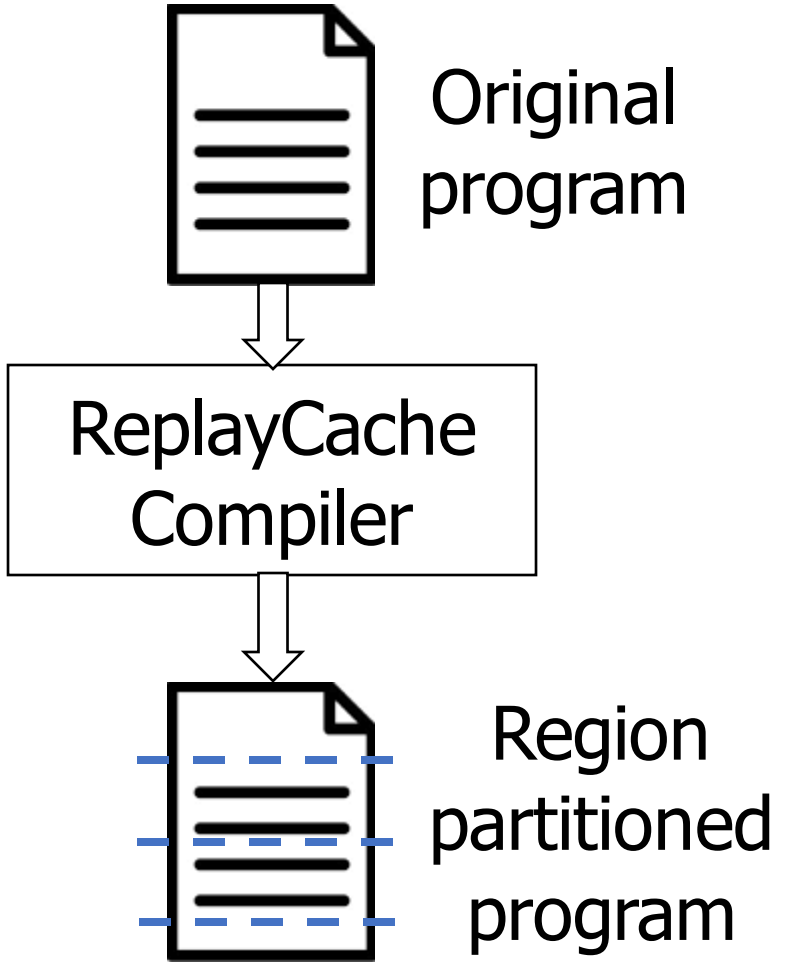




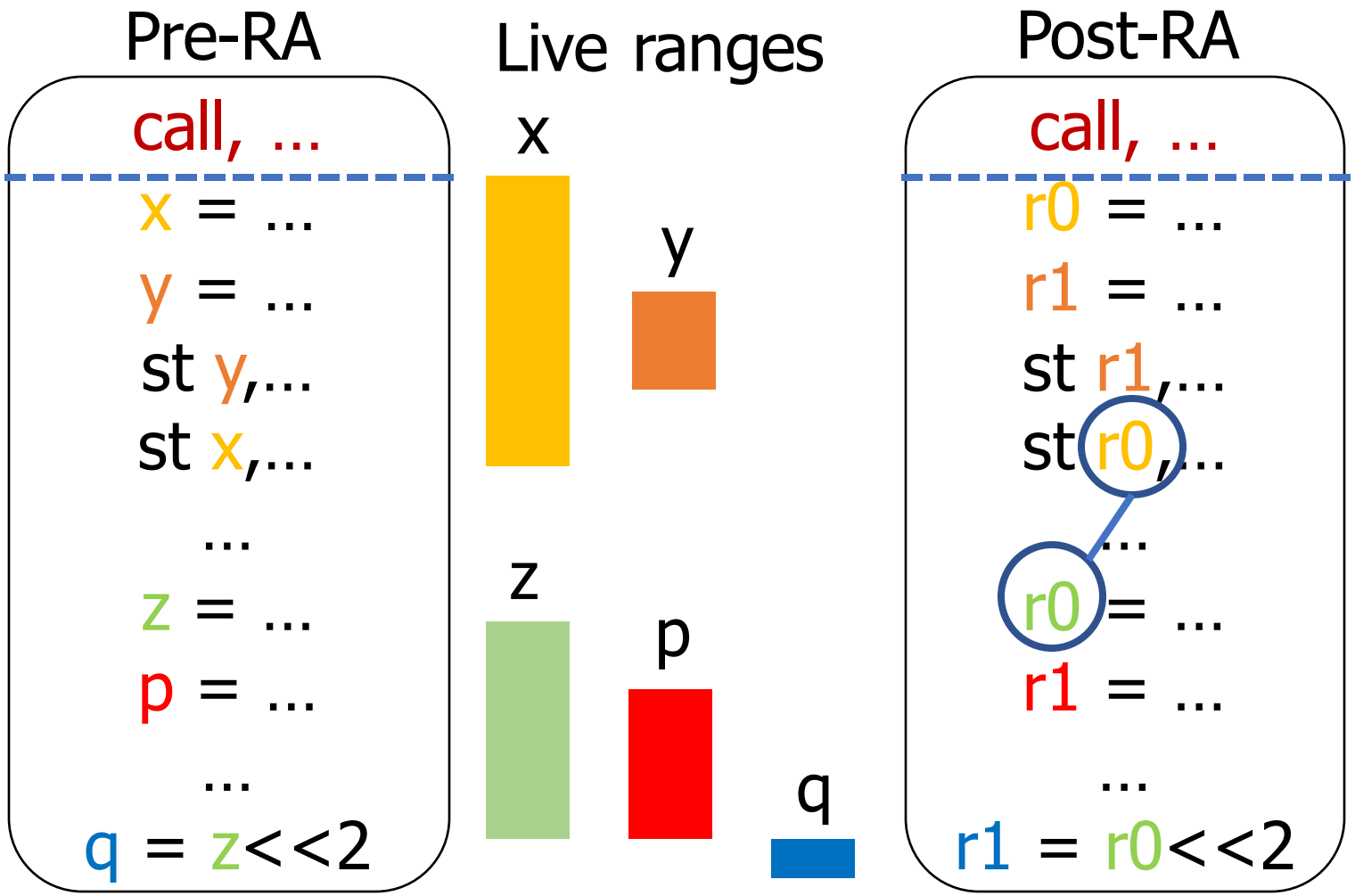
# Region-Level Recovery Protocol



# Challenge to Generate Store-Integrity Regions

# No Store Integrity with Existing Register Allocation




- Assume only 3 registers
- Disjoint live ranges can share the same physical register (x,z)

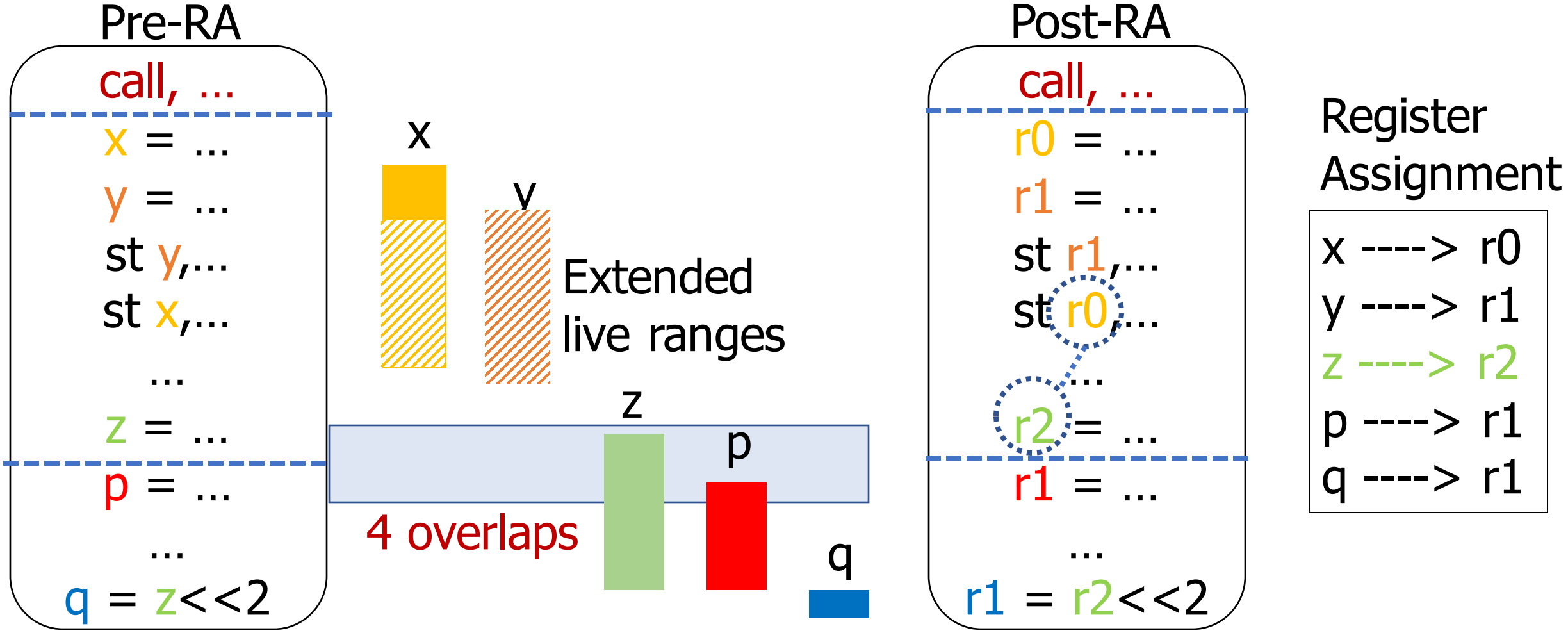
Register Assignment

```
x ----> r0
y ----> r1
z ----> r0
p ----> r1
q ----> r1
```

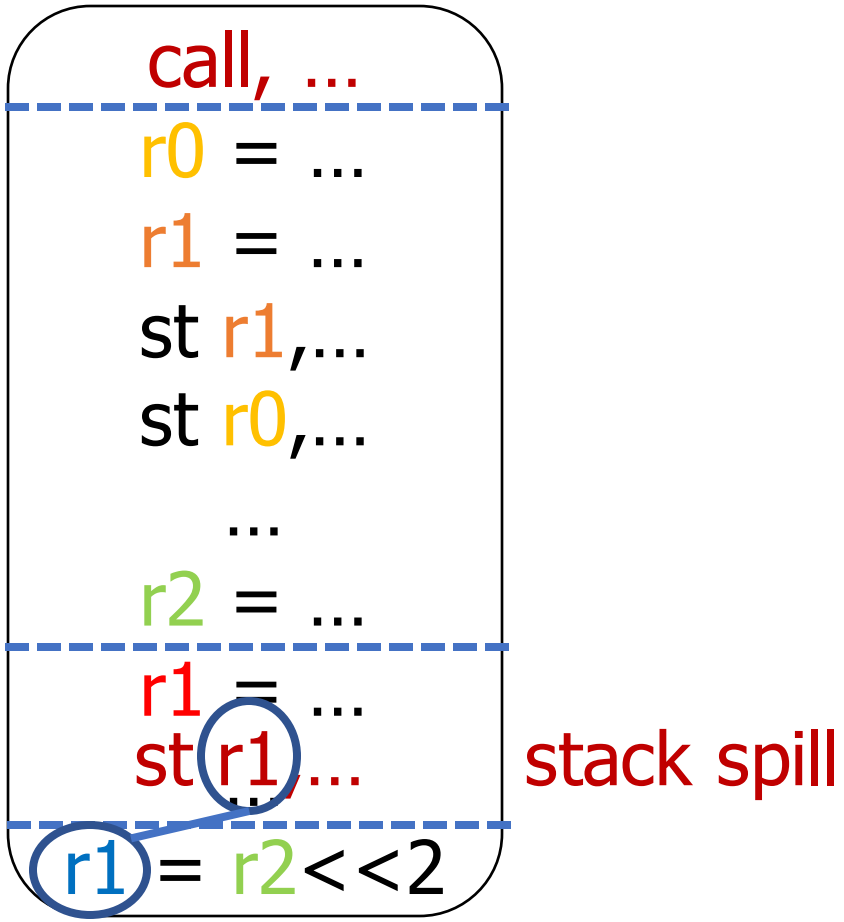
# Register-Renaming-Aware Region Partitioning

PURDUE UNIVERSITY

- Store operands must not share same registers with following definitions
- Assume 3 registers



# Spilling-Store-Registers Preservation




Register Assignment

x ----> r0
y ----> r1
z ----> r2
p ----> r1
q ----> r1

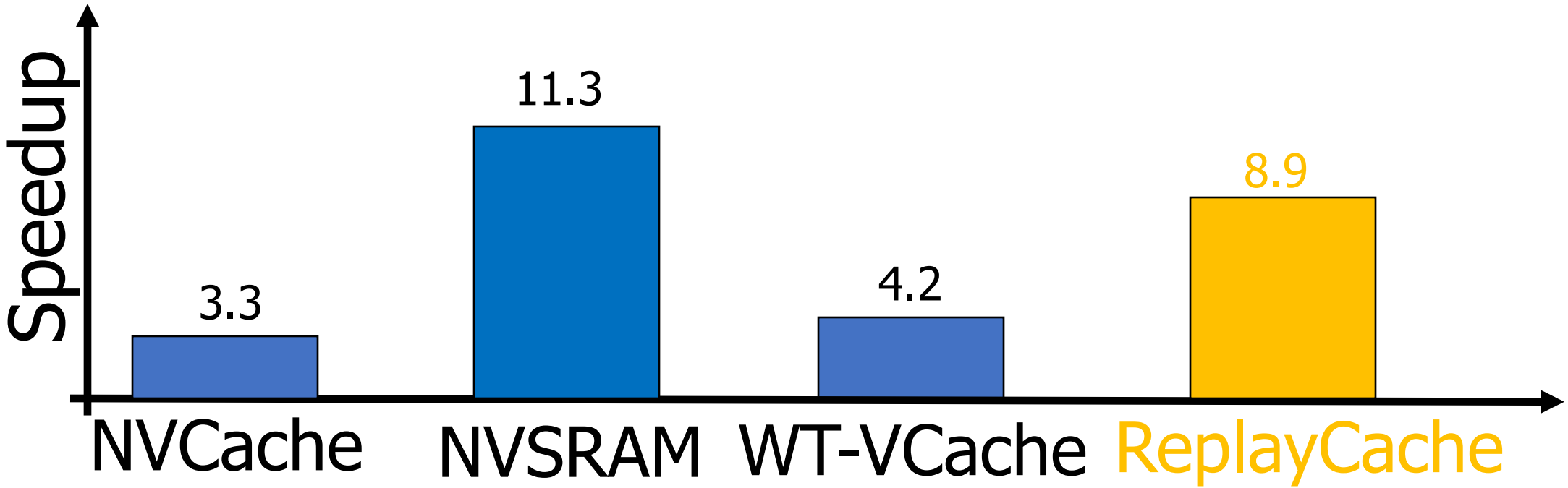
# Methodology and Evaluation

PURDUE UNIVERSITY

- Gem5-based NVPSim modeling a single core in-order ARMv7 processor with 8kB 2-way set-associative L1 I/D cache, and 16MB ReRAM as main memory.
- LLVM-based region formation.
- Mediabench + Mibench.

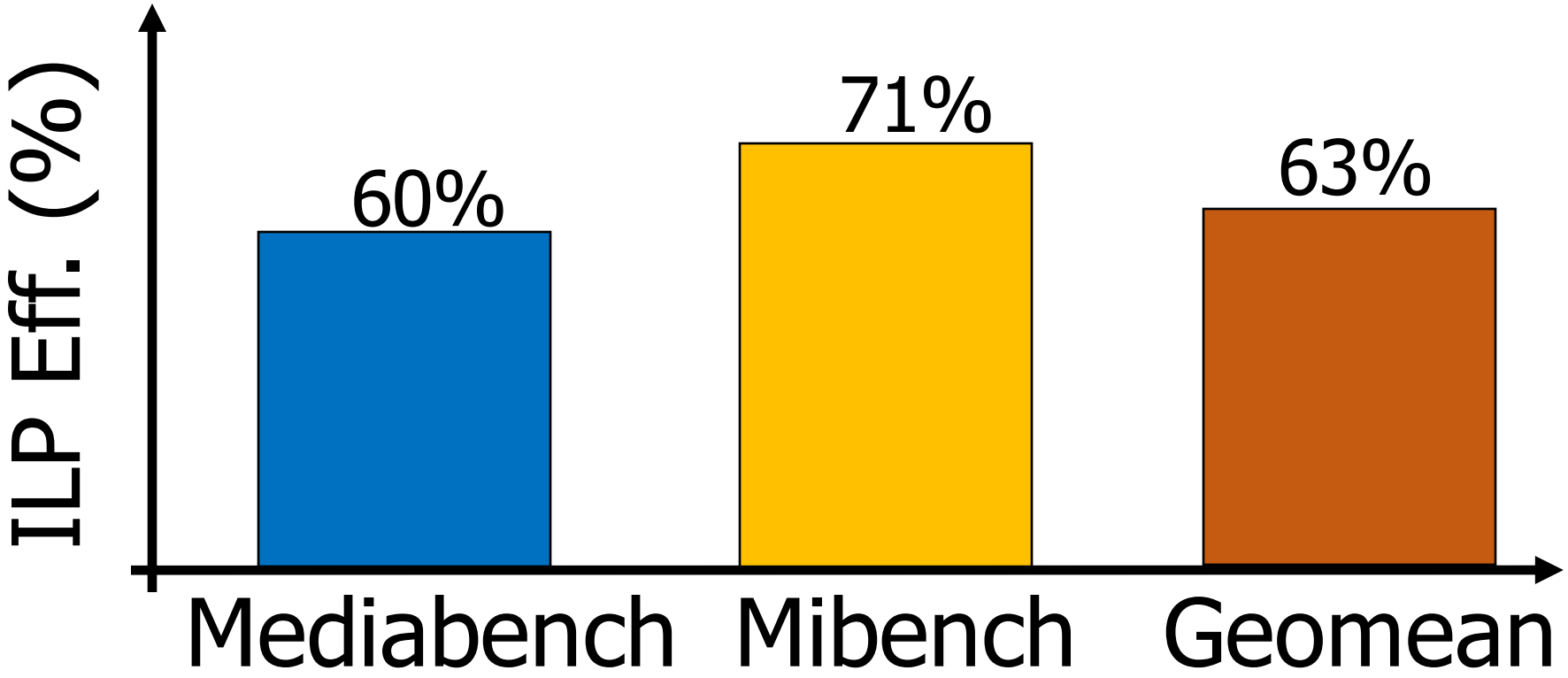
# Speedup over No-cache Baseline with Real Power Trace

PURDUE UNIVERSITY

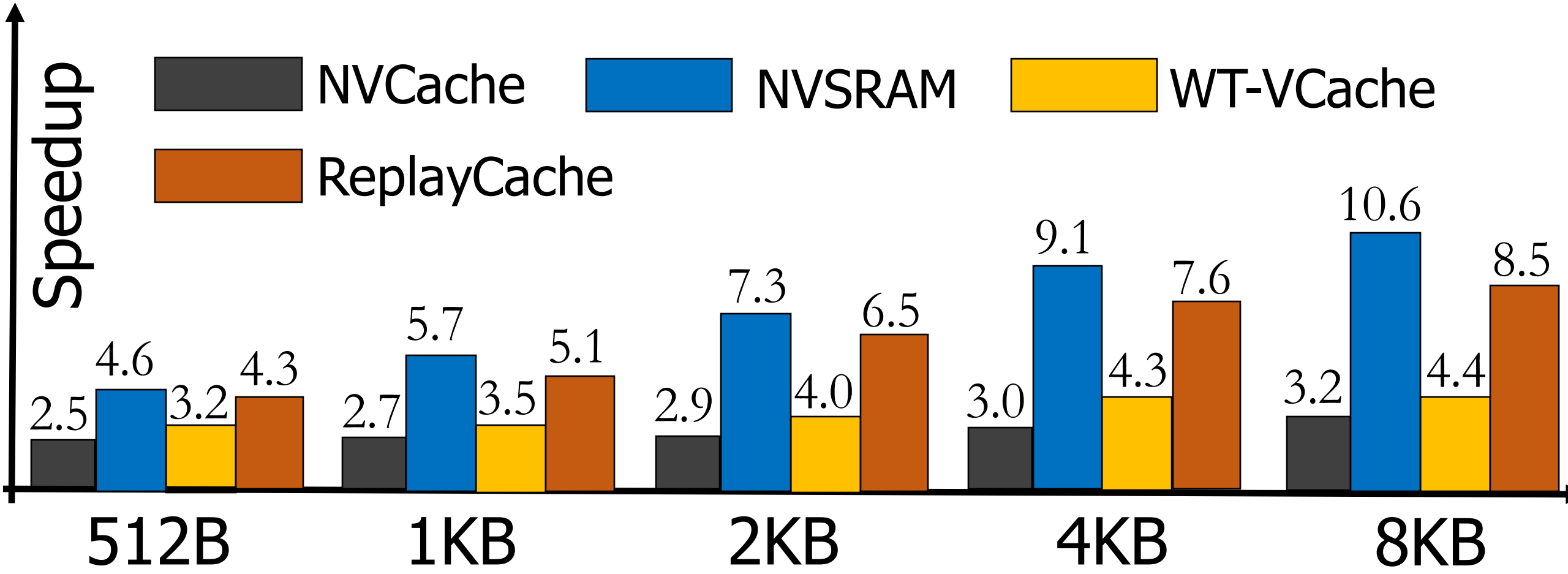


# ILP Efficiency (no power failure)

PURDUE UNIVERSITY



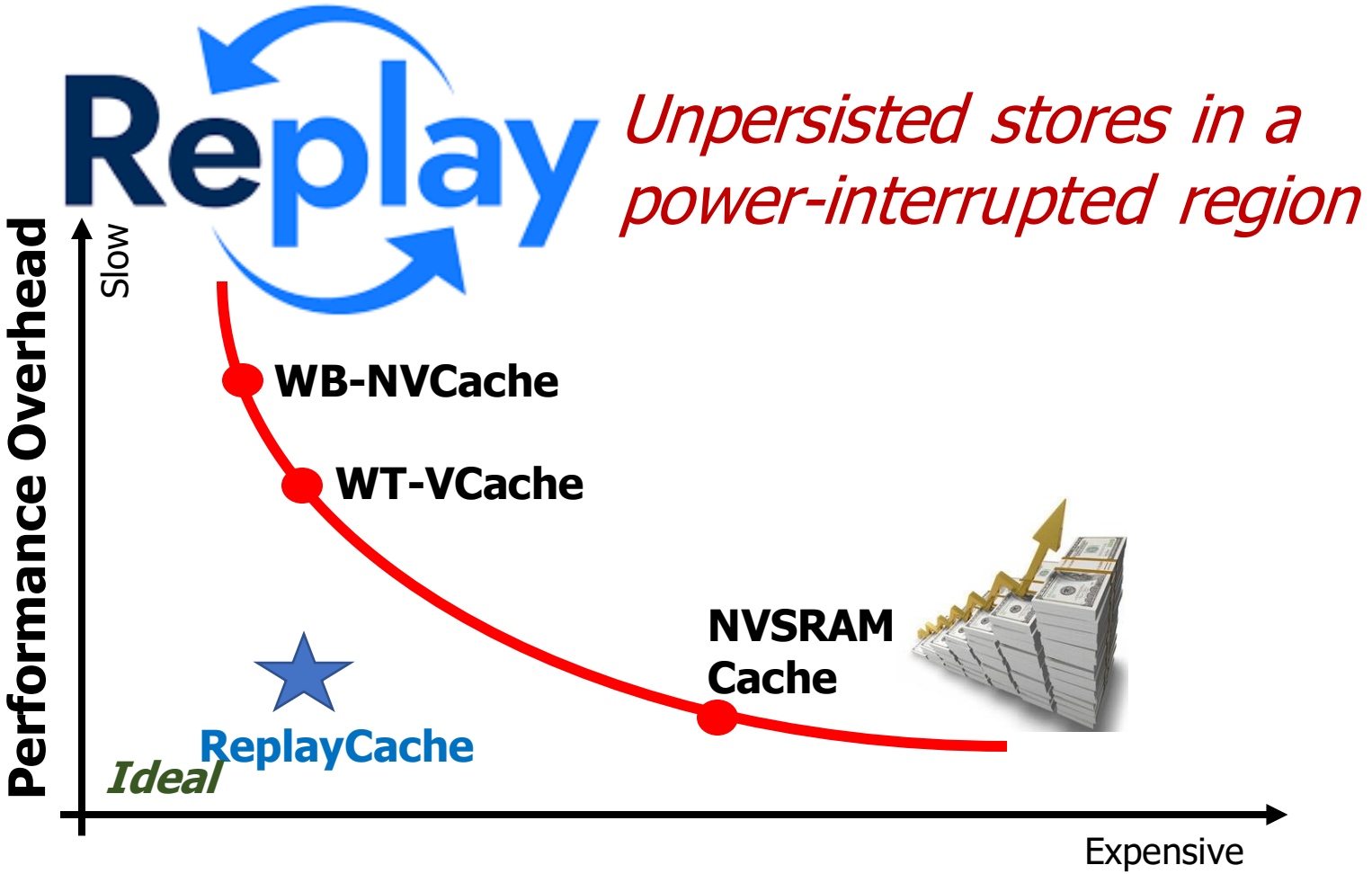
# Sensitivity to Cache Size




* Office power trace

# Conclusion

- A pure software design for enabling WB volatile cache with crash consistency guarantee.
- Never amplify writes.
- Comparable performance to an ideal NVSRAM cache for realistic power traces.



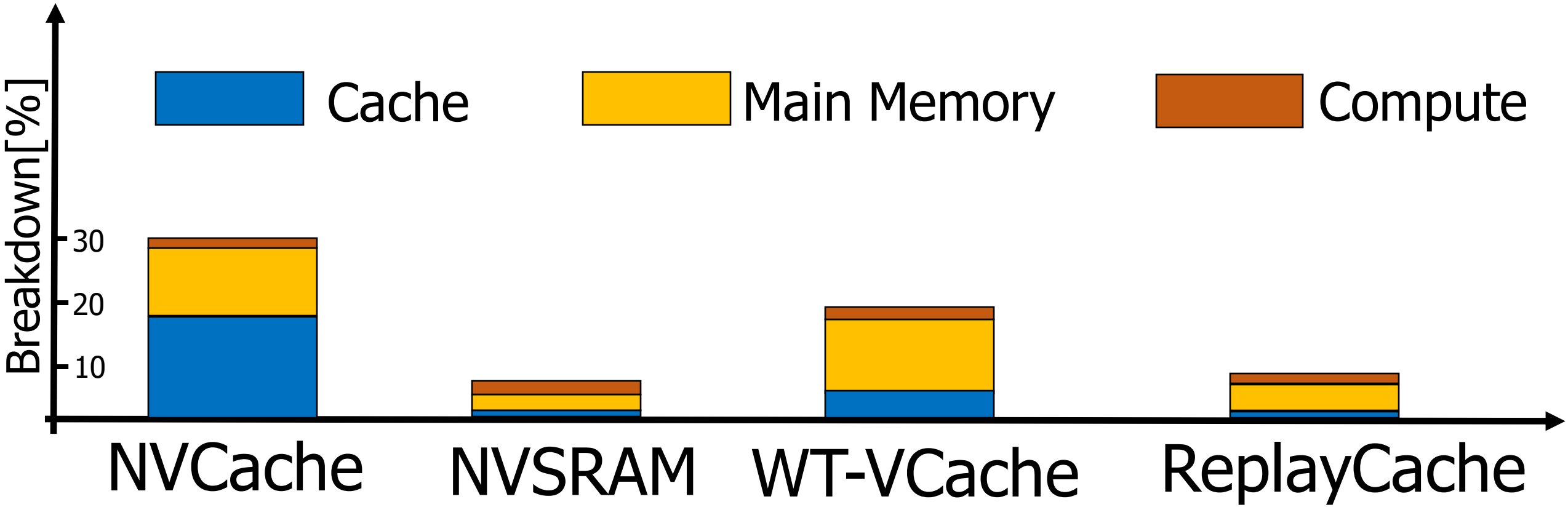
PURDUE UNIVERSITY

# Thank You
# Q&A

PURDUE UNIVERSITY

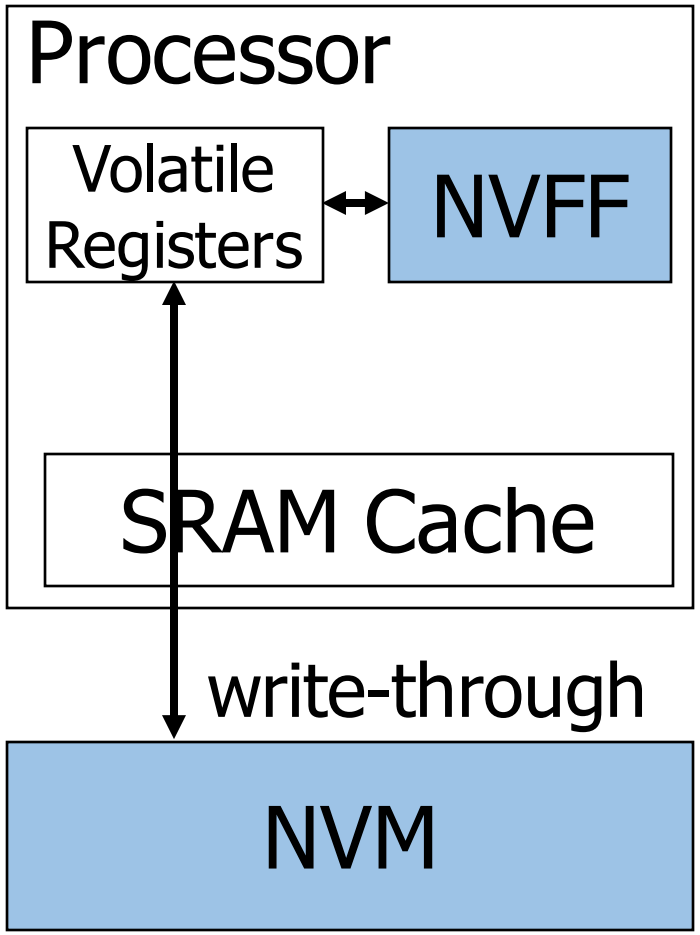
Jianping Zeng @Purdue University
(zeng207@purdue.edu)

# Normalized Energy Consumption Breakdown

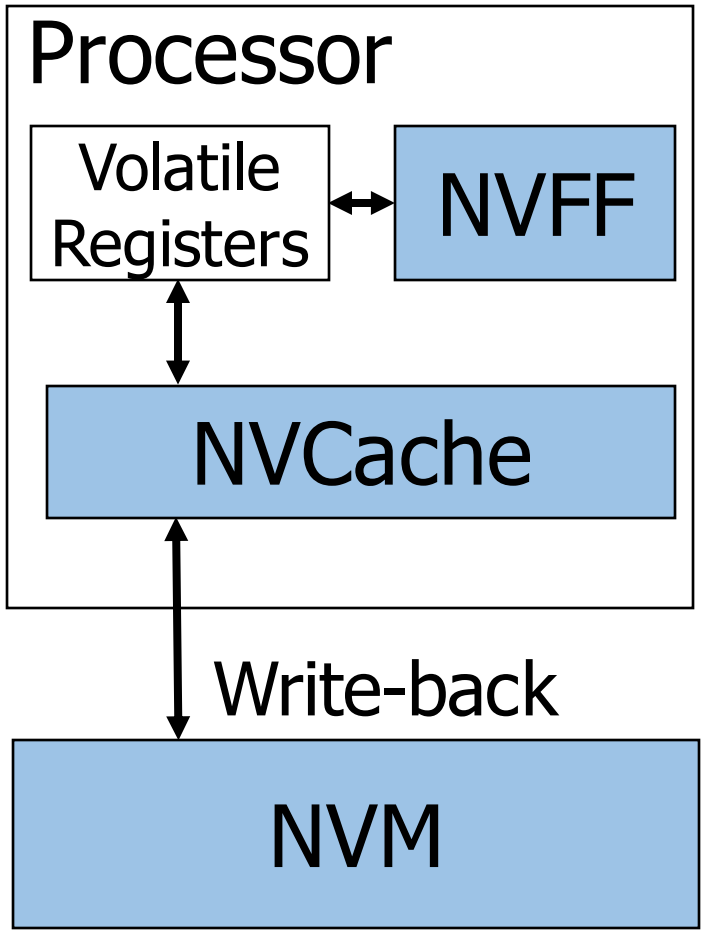



* Office power trace

# Architecture of WT-VCache and WB NVCache




(a). Write-through (WT) volatile cache




(b). Write-back (WB) Nonvolatile cache

# Speedup over Non-cache Baseline