# Reducing SSD Read Latency via NAND Flash Program and Erase Suspension

**Guanying Wu and Xubin He**

**{wug, xhe2}@vcu.edu**

Department of Electrical and Computer Engineering

Virginia Commonwealth University

Richmond, VA

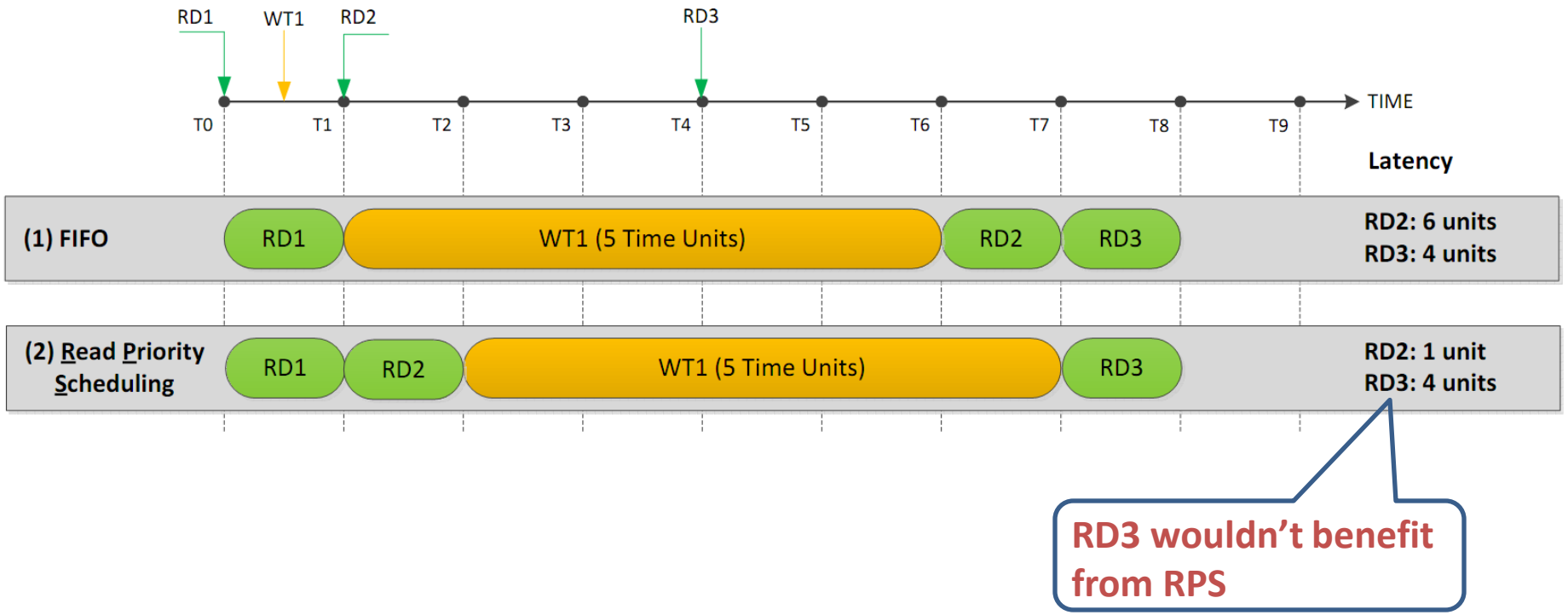

This research is sponsored in part by the U.S. National Science Foundation (NSF) under grants CCF-1102605 and CCF-1102624.

Suspend Program and Erase to make Read faster?

# Motivation

- P/E (Program/Erase) are about 10x/100x slower than read.
- P/E are non-suspendable in current NAND products
  - Once committed to NAND flash, no preemptions.
- If apps write and read at the same time & Intensive workloads
  - Read latency suffers from queuing delay.

# A Simple Demo

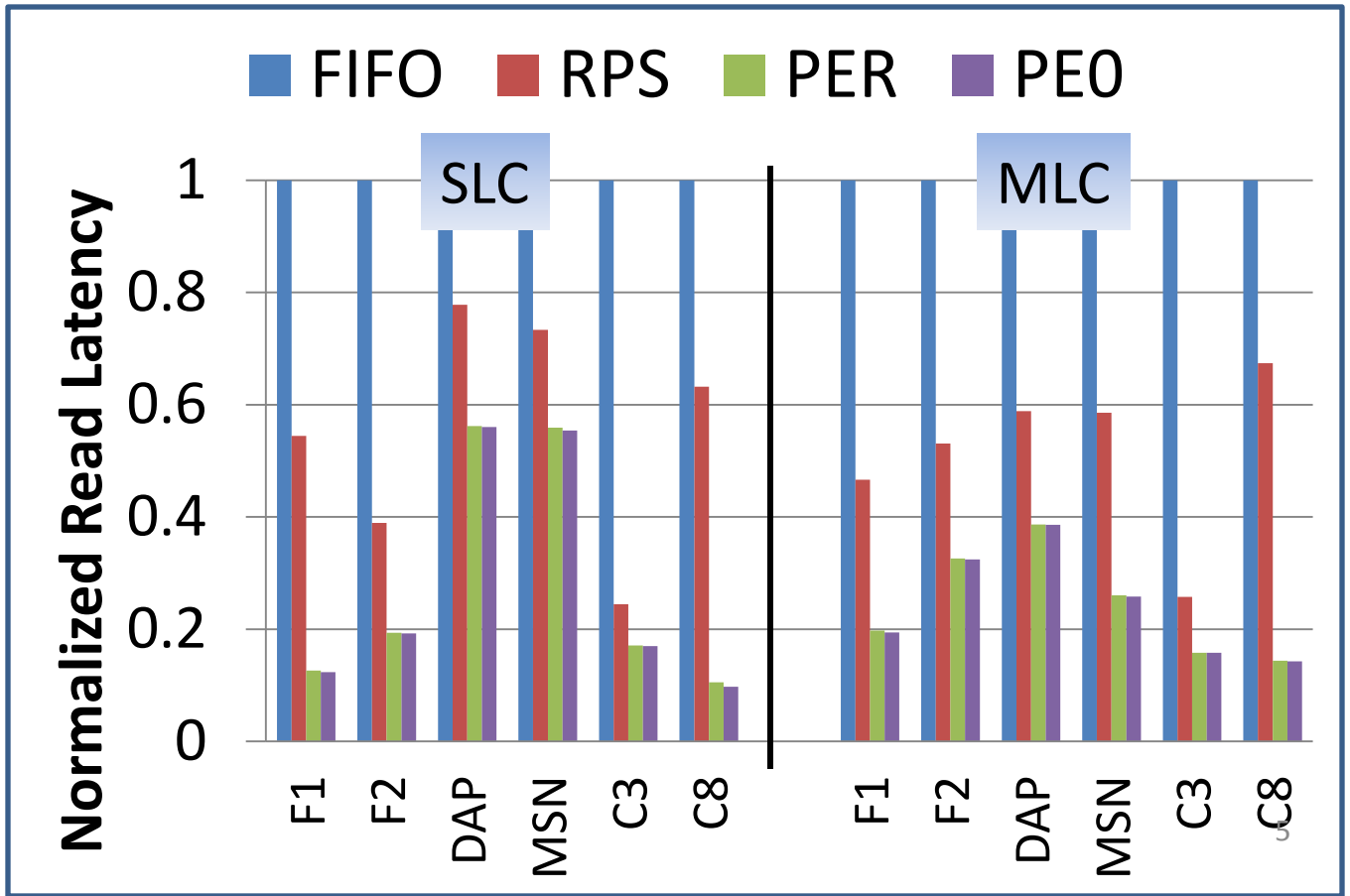


# Further Investigation

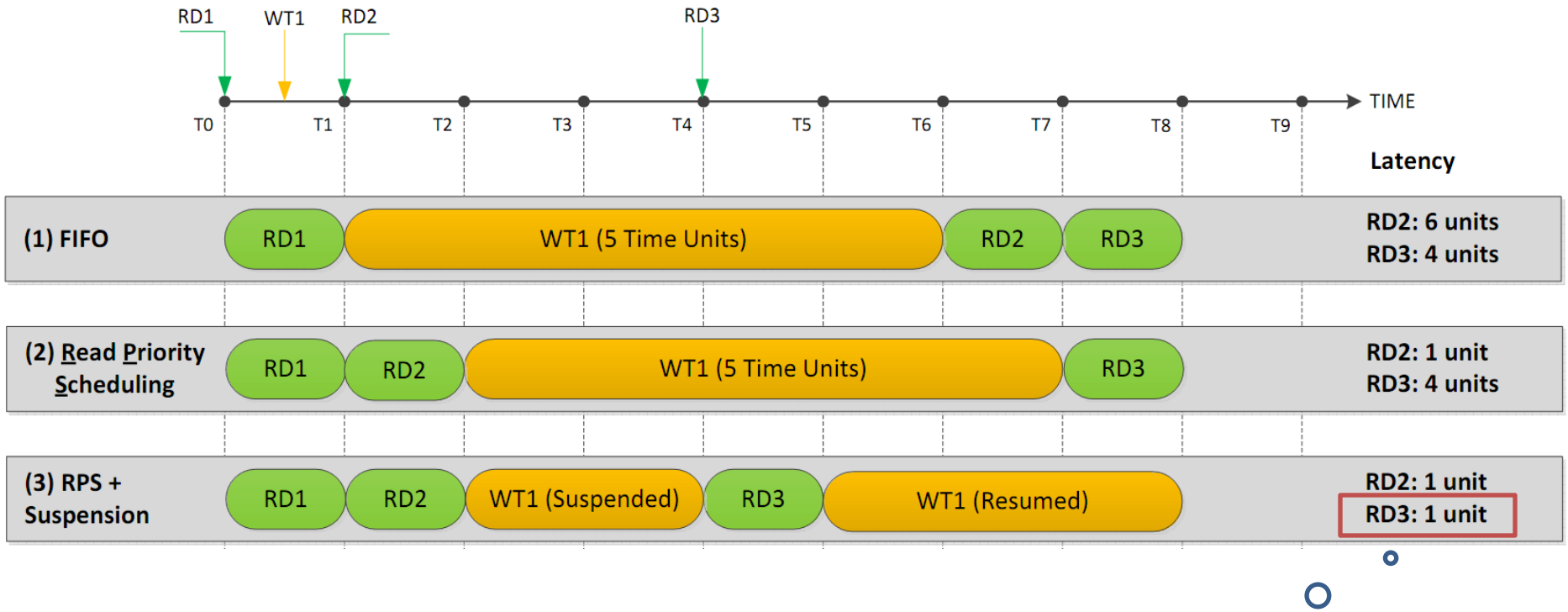
- Simulation with disk I/O traces.
  - MS SSD-add-on simulator
  - 6 popular traces

- Comparing:
  - FIFO
  - Read Priority Scheduling
  - Optimistic cases:
    - Equal latencies: PER
    - **0** P/E latencies: PE0

RPS isn't Good Enough



# Our Idea: **Make NAND Flash P/E <u>Suspendable</u>**



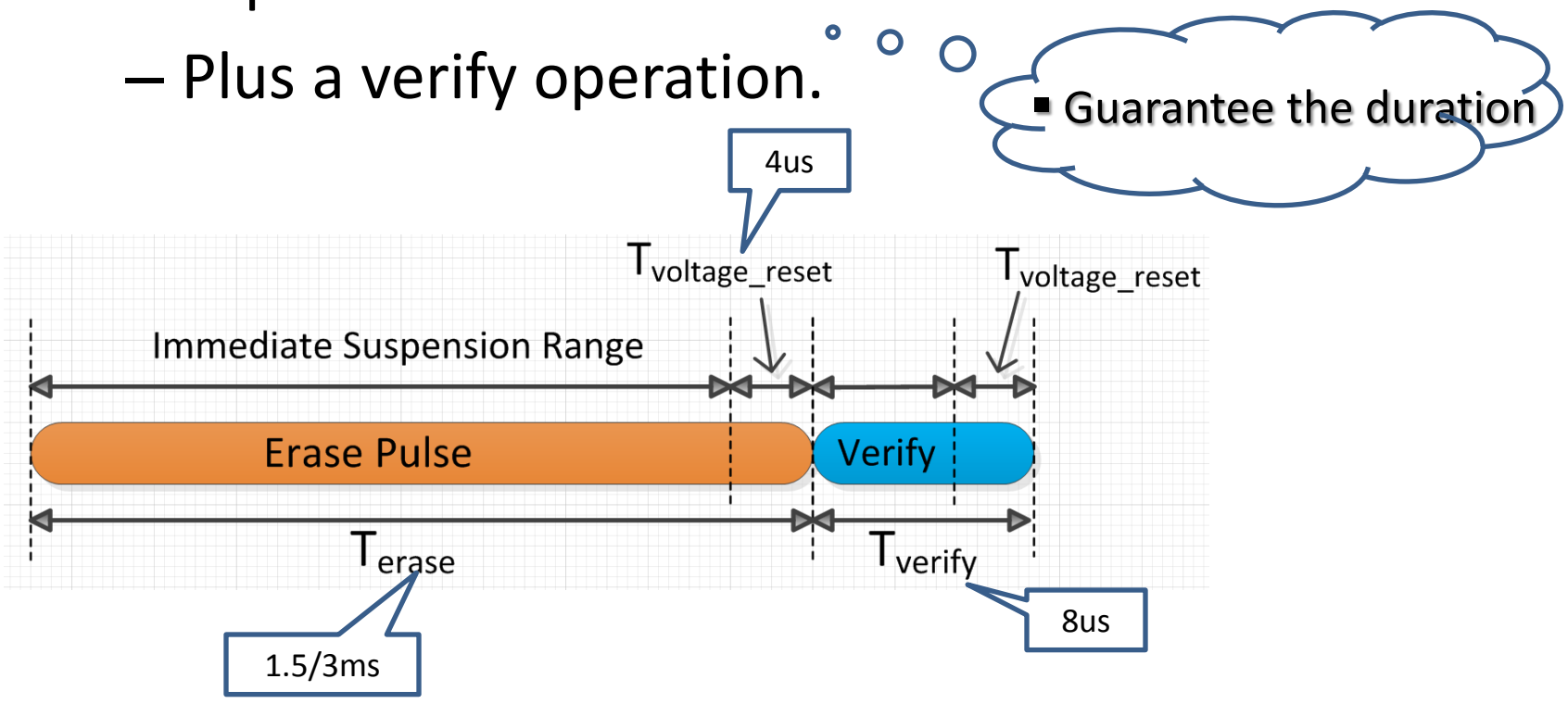

We are expecting:

# Outline

- Background:
  - Why can we suspend P/E?
- Design:
  - How do we do it?
- Evaluation:
  - Compare to the optimistic cases
- Conclusion

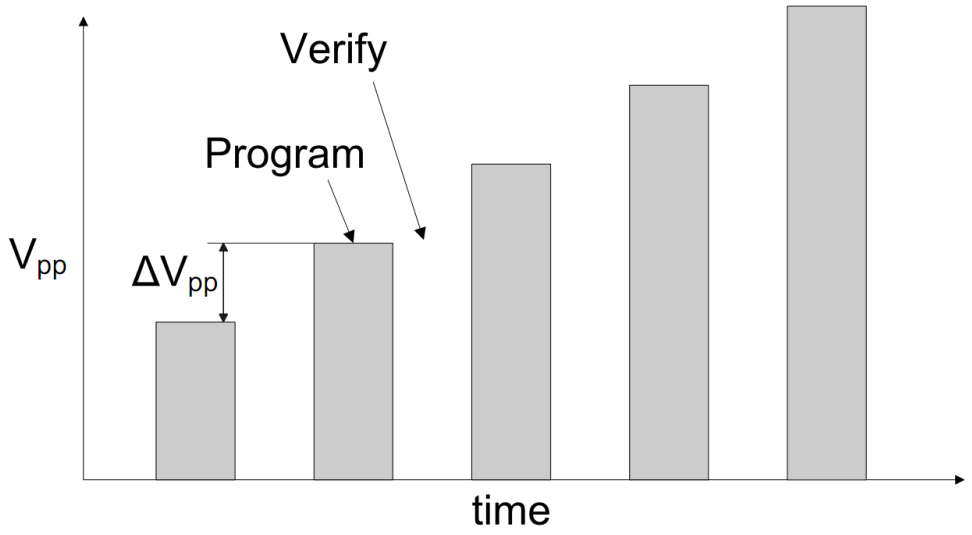
# Background: NAND Flash Erase

- NAND Flash Erase:
  - Reset cells via a **long pulse** of Erase Voltage to expel the electrons.
  - Plus a verify operation.



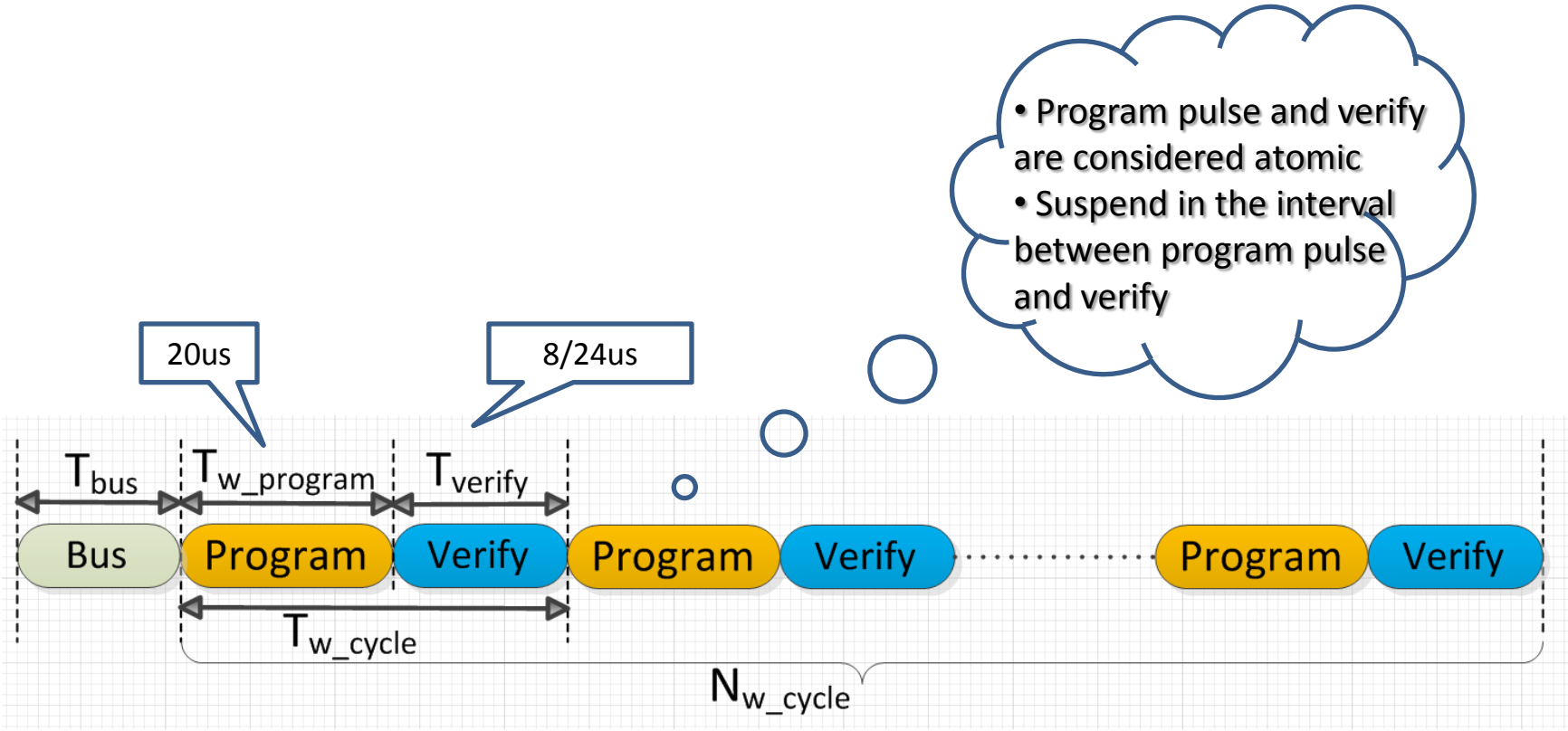
# Background: NAND Flash Program

- NAND Flash **Program**:
  - Incremental Step Pulse Programming




ARASE, K. Semiconductor NAND Type Flash Memory with Incremental Step Pulse Programming, Sept. 22 1998. U.S. Patent 5,812,457.

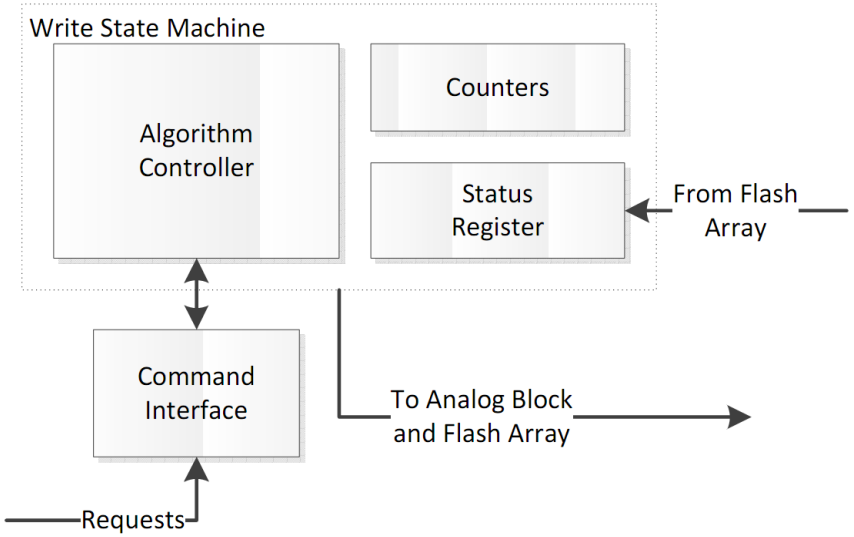
# Background: NAND Flash Program




NAND Flash Program: Timeline

# Background: NAND Flash P/E

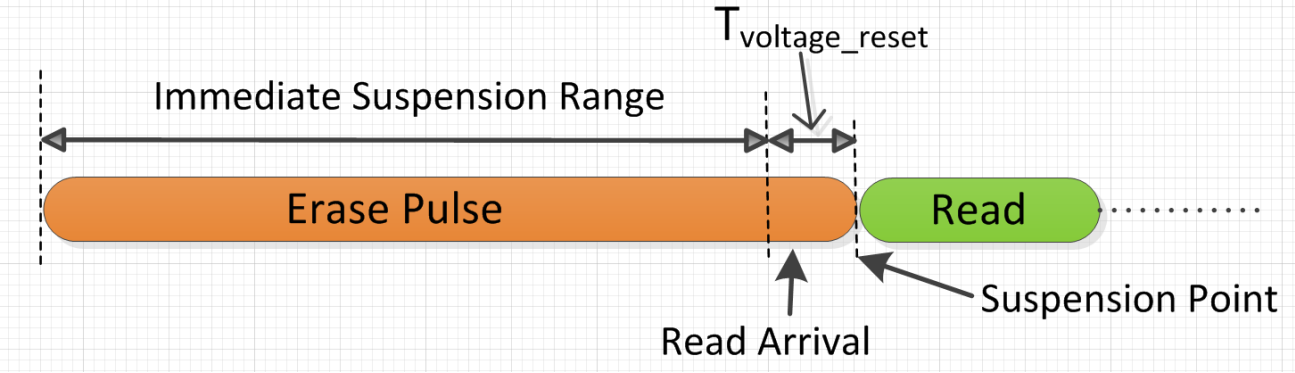
- Correct Timing
  - Program: what is the last phase? what is the value of Vpp?
  - Erase: how much job have we done/how much is left?



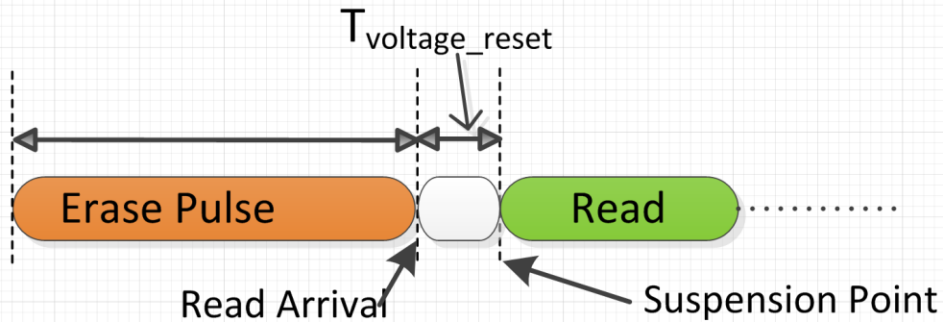

BREWER, J.; GILL, M. *Nonvolatile Memory Technologies with Emphasis on Flash*.

# Design: Suspend/Resume Erase

- Case 1: Read arrives when resetting wire voltage




- Case 2: Read in the middle of Erase Pulse or Verify (cancelled)



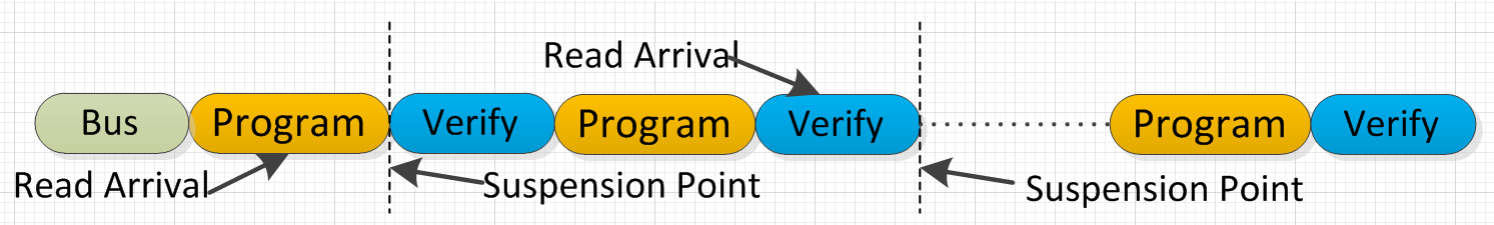
# Design: Suspend/Resume Erase

- Case 1: Suspension happens in **Verify** phase
  - Redo Verify phase. (overhead to erase latency)

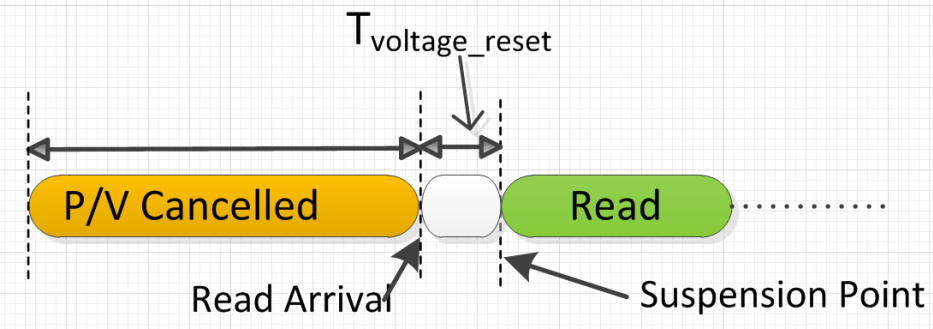
- Case 2: Suspension happens in **Erase** phase
  - Finish what is left before suspension.

# Design: Suspend/Resume Program

- Program pulse and Verify are considered atomic, intuitively:
  - Choice 1: Suspend in the intervals – **I**nter-**P**hase-**S**uspension

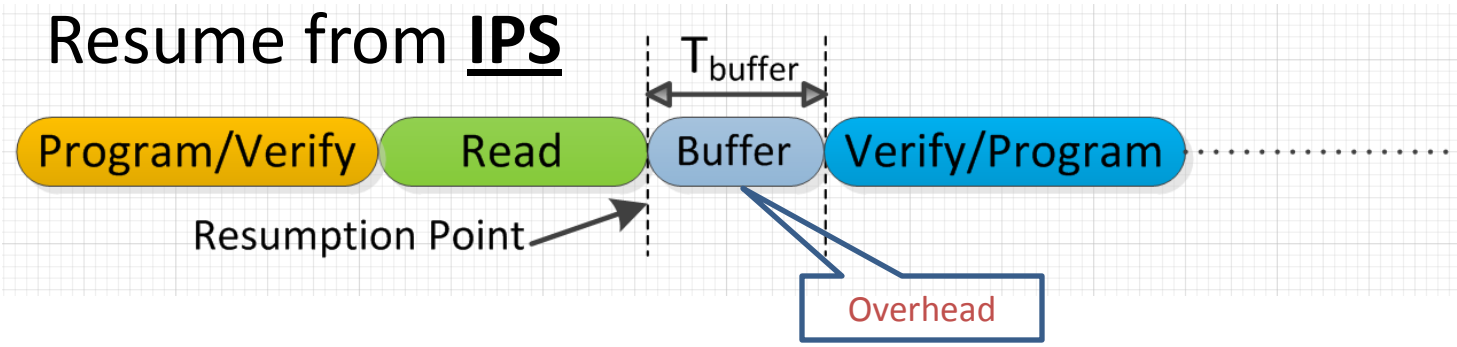



  - Choice 2: Cancel the current phase – **I**ntra-**P**hase-**C**ancellation

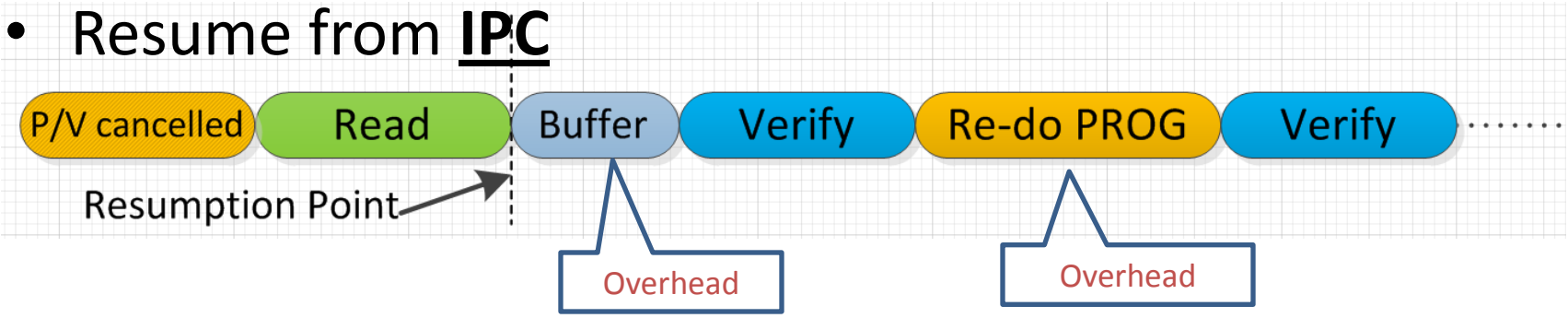


# Design: Suspend/Resume Program

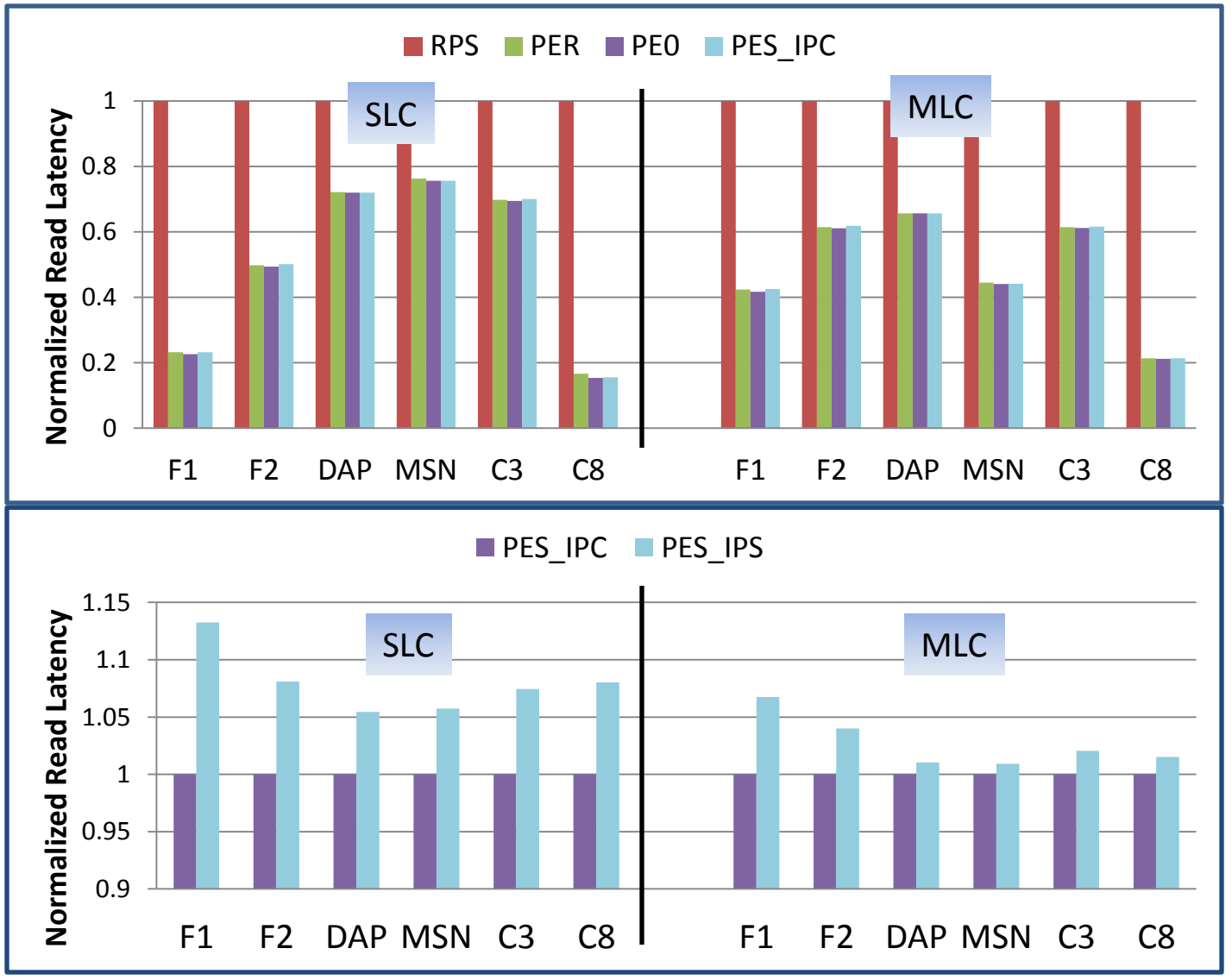
- Need to retain the page buffer first.
- Resume from **IPS**



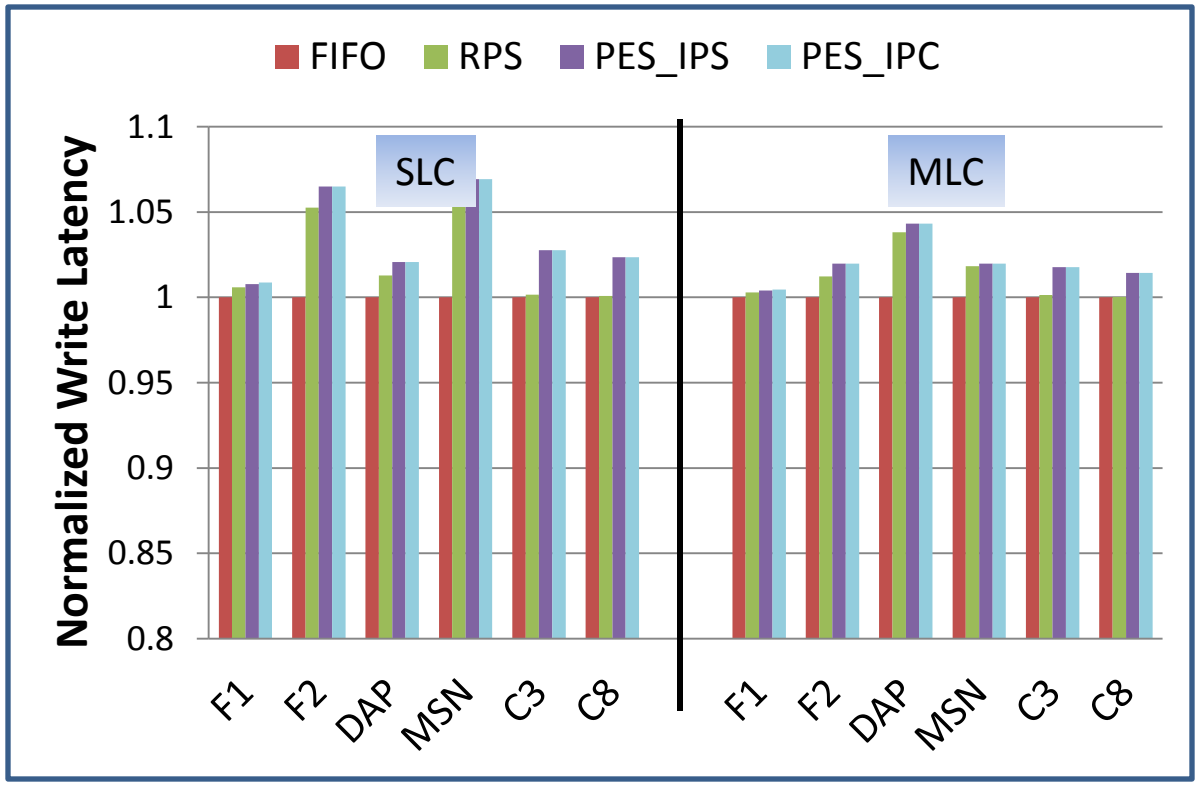

- Resume from **IPC**



# Evaluation: Read Latency



# Evaluation: Write Latency Overhead



# Conclusion

- Suspending P/E for read is a feasible solution:
  - Significant read performance gain.
  - Low overhead on write latency.

**Thank You!**

Questions?