STORAGE DEVELOPER CONFERENCE

SDC 23

BY Developers FOR Developers

Live Migration for PCIe® SSDs

Presented by Dan Helmick, PhD

SAMSUNG

Agenda

- Background and Assumed System Set-up
- Pre-Copy Phase: Start
- Pre-Copy Phase: Namespace (NS) Migration
- Stop-and-Copy Phase: Pause and Final Copies

SAMSUNG

SDC 23

Live Migration Background

- This presentation focuses on the SSD aspects of implementing Live Migration in a direct attached scenario.

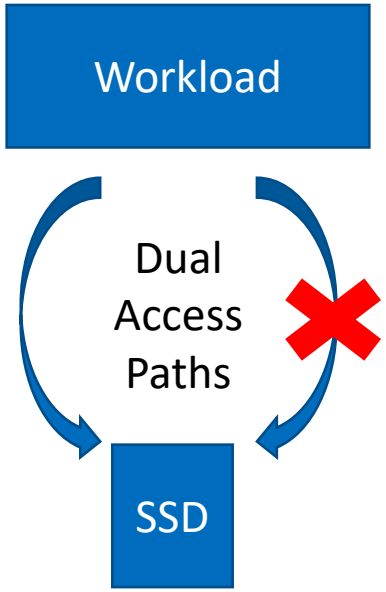
- NVMe Resources
  - TP4165 Tracking LBA Allocation with Granularity
  - TP4159 PCIe® Infrastructure for Live Migration
  - TP4176 Quality of Service for NVM subsystem Resources for a Controller

- Public Conference Resources
  - Flash Memory Summit Presentation “Host Controlled Live Migration” by Mike Allison and Lee Prewitt
  - Storage Developers Conference “NVM Express® State of the Union” by Ross Stenfort and Mike Allison
  - Open Compute Global Summit “Standardizing Live Migration with NVM Express®” by Mike Allison, Amber Huffman, and Lee Prewitt

SAMSUNG

SDC 23

Motivation for Live Migration



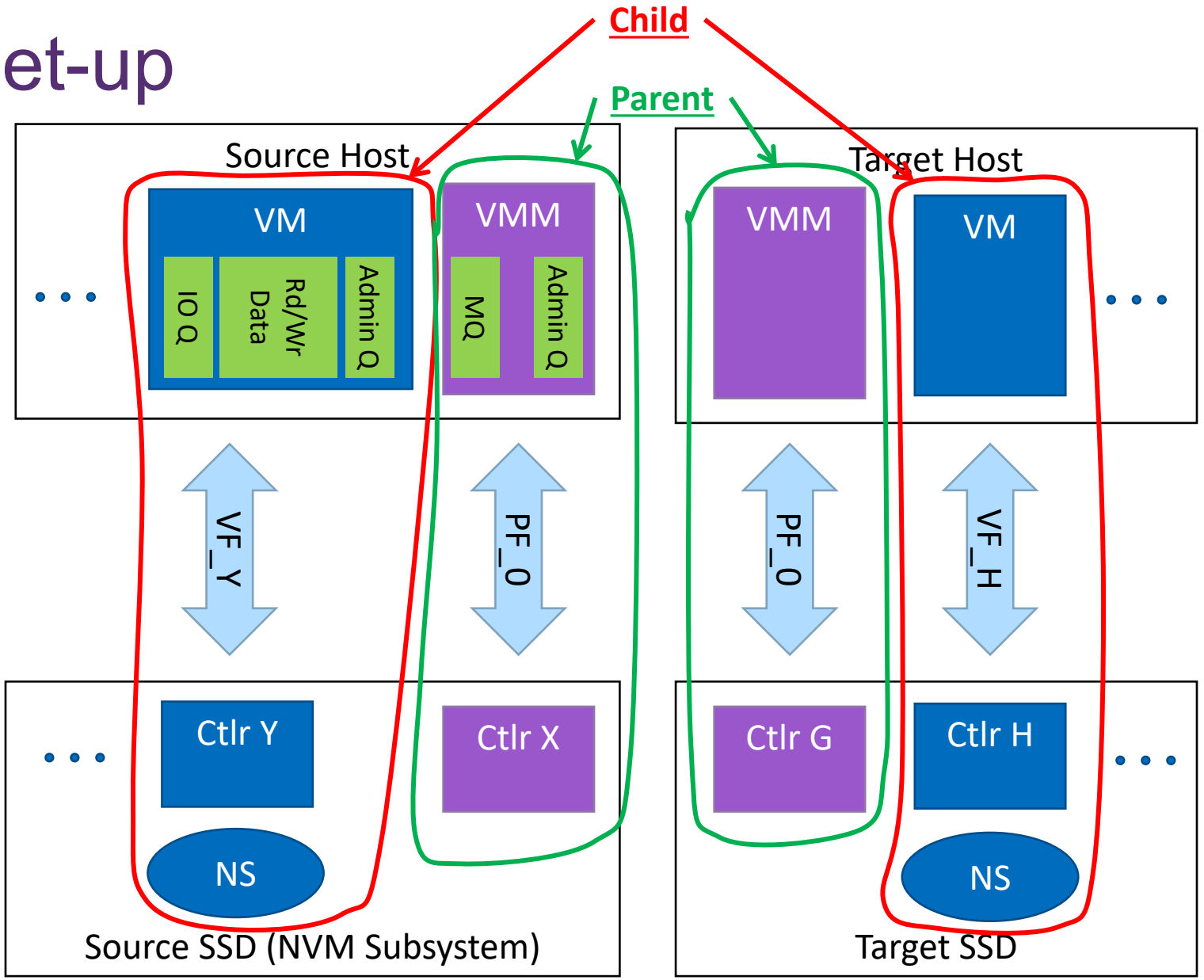
- Why Migrate a workload?
  - Data Center down time, errors, or other access anomalies
  - Load Balancing
    - Example:
    - AI training is long running without user interactions
    - Data Center's load may vary as a function of the local time zone
    - Migrate the AI training to a Data Center (DC) experiencing reduced load due to night time
- Why **Live** Migrate?
  - Workload can continue to run without awareness of migration event
  - Minimizes downtime
- Why enable Live Migration at the SSD?
  - Allows the removal of SW shim layers on the IO queues
  - Reduces Host SW load
  - Improved storage access latencies

SAMSUNG

SDC 23

An Example System Set-up

- Virtual Machines (VMs) and VM Monitor (VMM)
  - 1 VMM to many VMs
  - All Live Migration (LM) commands come through VMM
  - May not share memory spaces
    - Ex: Migration Queue (MQ) in VMM memory space
    - Ex: VM's IO and Admin Queues in VM memory space
  - VM is unaware LM is happening
  - Logging in the MQ may be in the form of Migration Queue Entries (MQE)
- SSD example with SR-IOV
  - Primary Controller (Ctlr) per VMM on PF_0
  - Secondary Ctlr per VM on VF_Y and VF_H
- Target vs Source
  - Similar setups
  - Target VM may send writes/reads to Ctlr H prior to "start"
    - Target VM's commands may be generated by VMM prior to migration

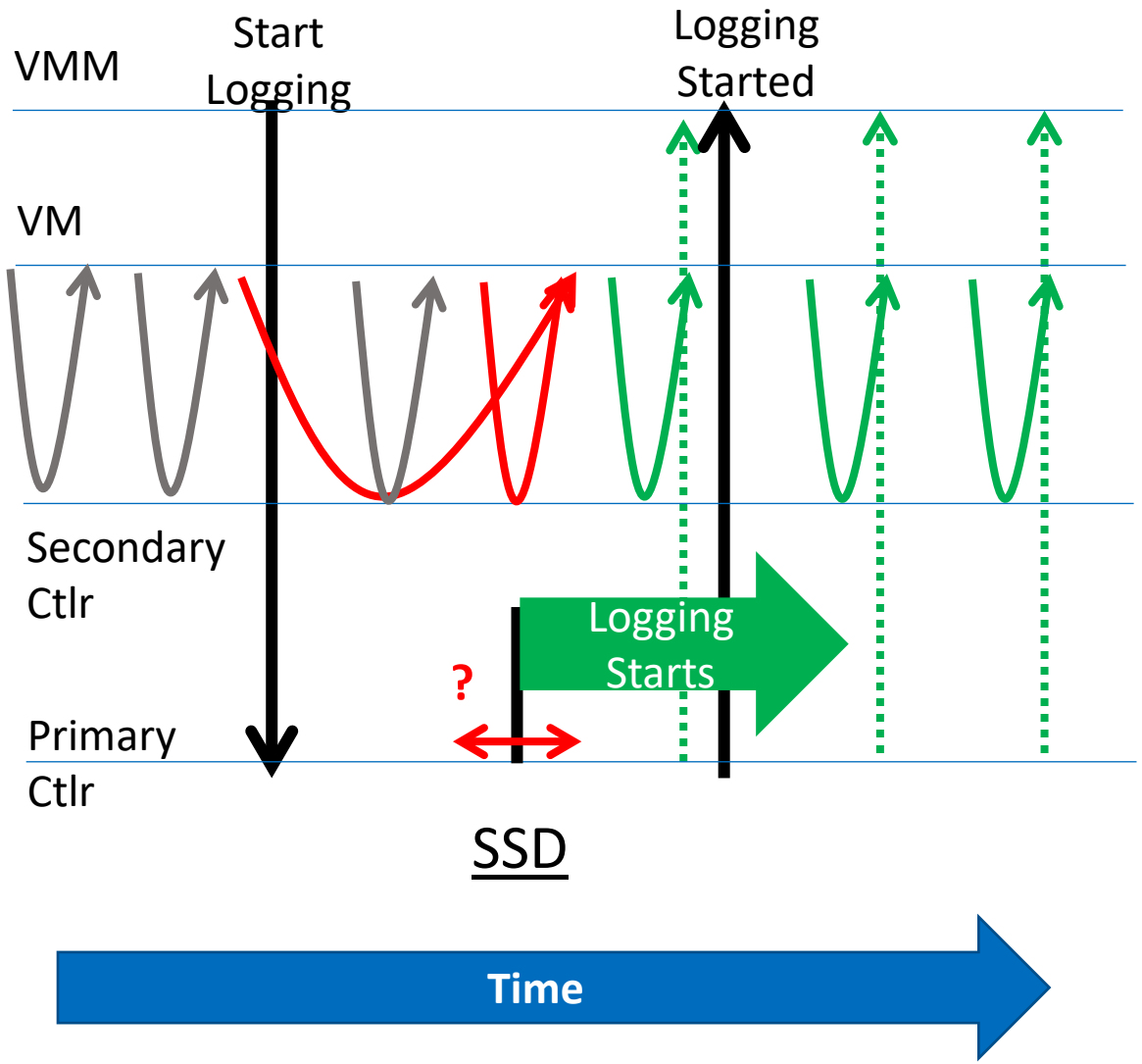


SAMSUNG

SDC 23

Pre-Copy Phase: Start Logging

- VM continues to interact with Secondary Ctlr on SSD (Rd/Wr)
  - **Race Conditions** are a concern
- "Start Logging" Command Flow
  - Ongoing VM IOs
  - VMM sends "Start Logging" Command
  - Primary Ctlr begins tracking all requested MQ events occurring in VM's Ctlr (Secondary Ctlr)
    - Some commands in flight may be logged (excess logging is allowed)
    - Some commands in flight may not be logged
  - Primary Ctlr completes "Start Logging" Command
    - SSD Promise: All potentially log-able commands will now be logged
- VMM has successfully started logging in MQ
  - Relationship of Logging Start and some commands is unknown
  - Unknown timing of where Logging Start occurred with respect to Completion of Start Logging command
  - "Logging Started" ensures
    - All prior commands in flight have finished
    - All future commands in flight will be logged

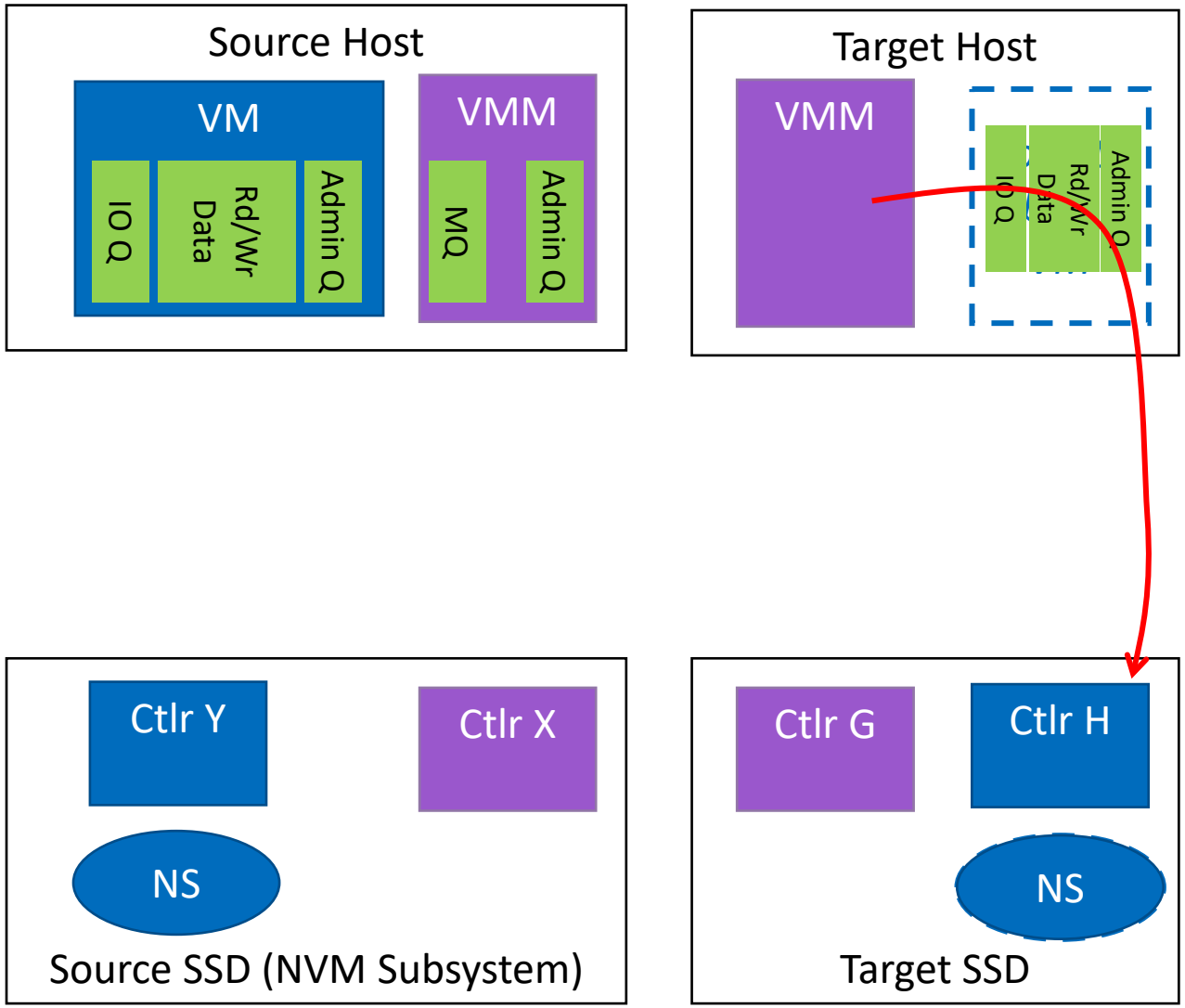


SAMSUNG

SDC 23

Pre-Copy Phase: Target Preparation

- Target Precondition
  - Available Secondary Controller
  - Available Host side VM resources
- Standard NVMe commands for initializing Target SSD
  - Initialize any Queue and IO command structures needed
  - Create NS
- Above illustrates one potential flow, but other options exist
  - Ex: Shared NS created by VMM on Ctlr G

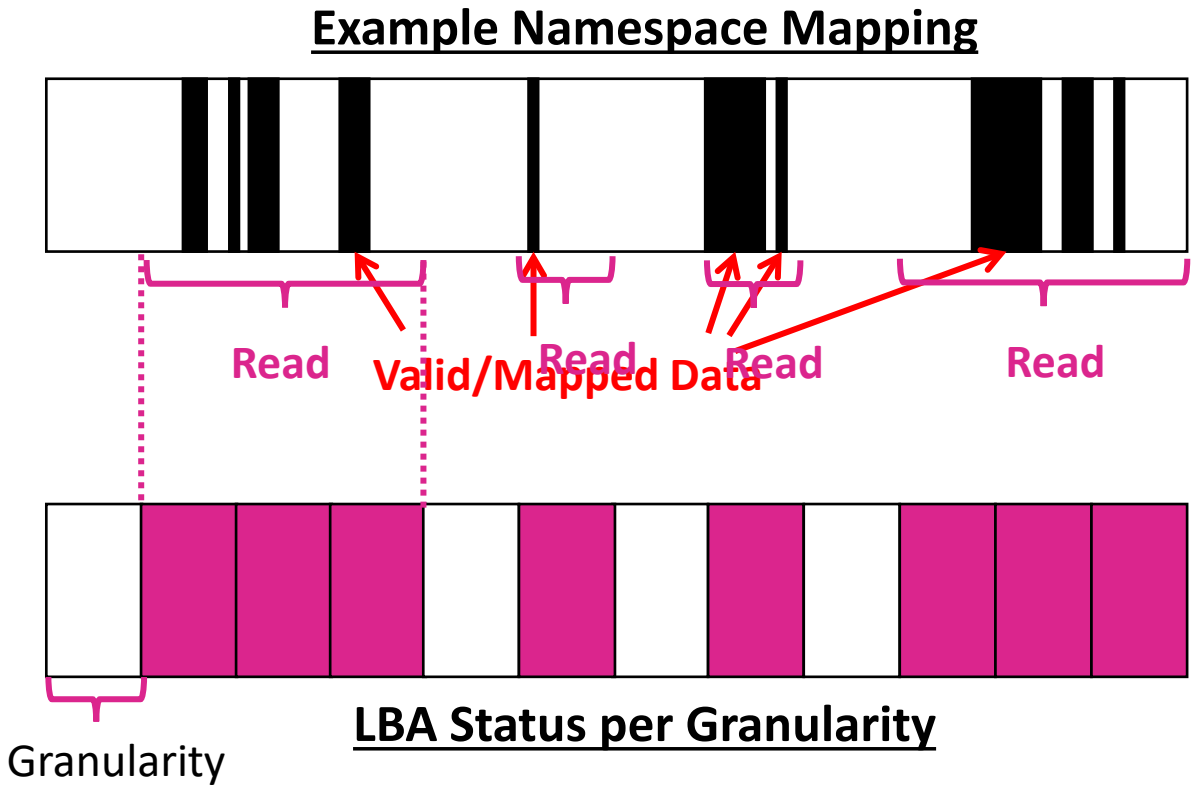


SAMSUNG

SDC 23

Pre-Copy Phase: Initial NS Migration

- Option 1:
  - VMM copies entire VM NS
    - Not optimal for sparsely written data
    - *<See example on right>*
- Option 2:
  - VMM sends Primary Ctlr: Get LBA Status
    - Granularity: Set by SSD
      - Customer requirements discussion
  - Primary Ctlr
    - Returns results with granularity restrictions
    - Any data state other than deallocated is returned as mapped
      - Ex: Read Uncorrectable
  - VMM
    - For each mapped LBA status
    - Submitted as Read of Child's NS



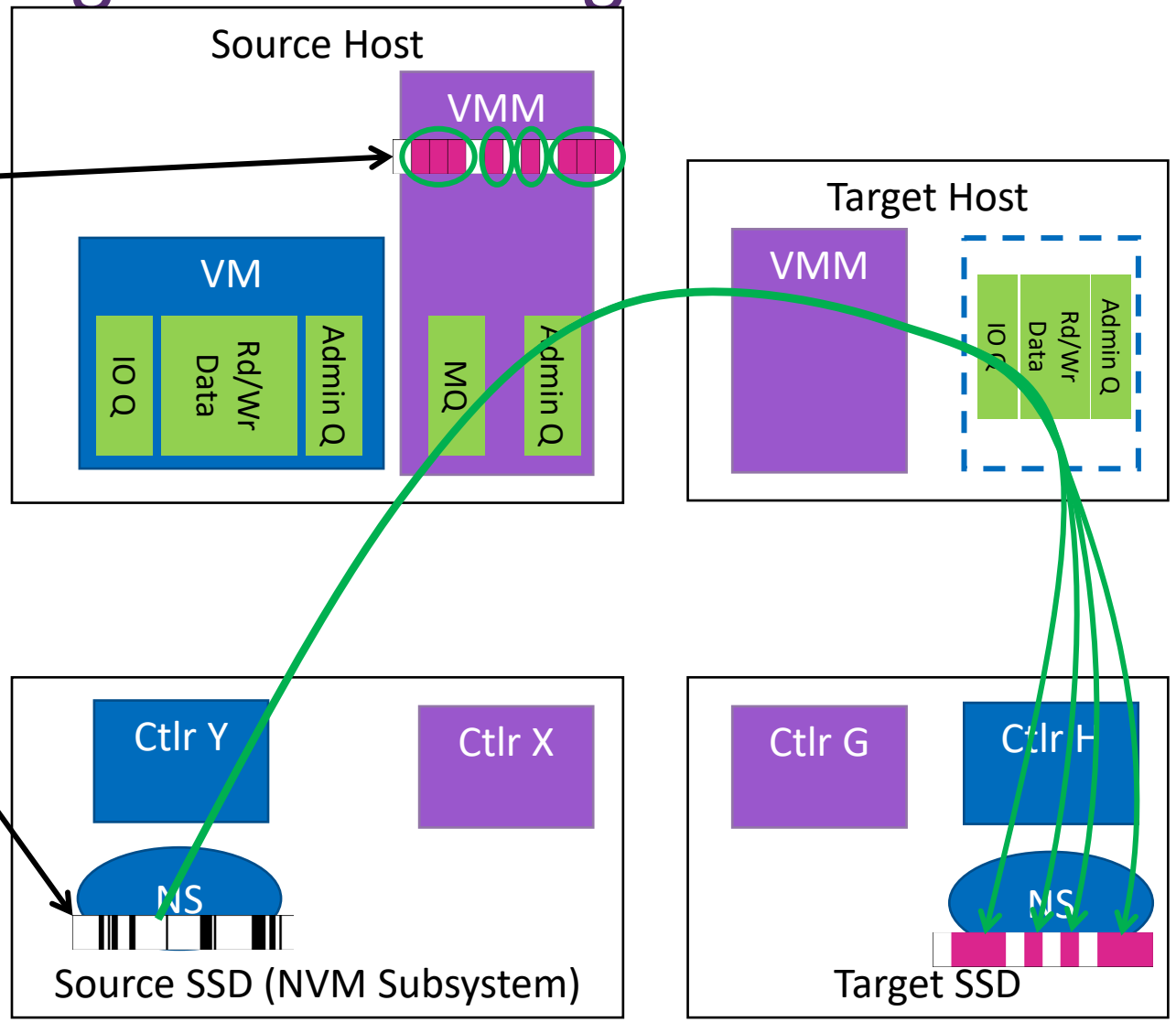
For more info: TP4165 Tracking LBA Allocation with Granularity

SAMSUNG

SDC 23

Pre-Copy Phase: Initial NS Migration to Target

- VM's NS Mapping
- Returned LBA Status per Granularity
- VMM submits Read to Child's NS for each contiguous mapped LBA range
- New NS is populated with no dependence/knowledge of Source SSD's granularities



SAMSUNG

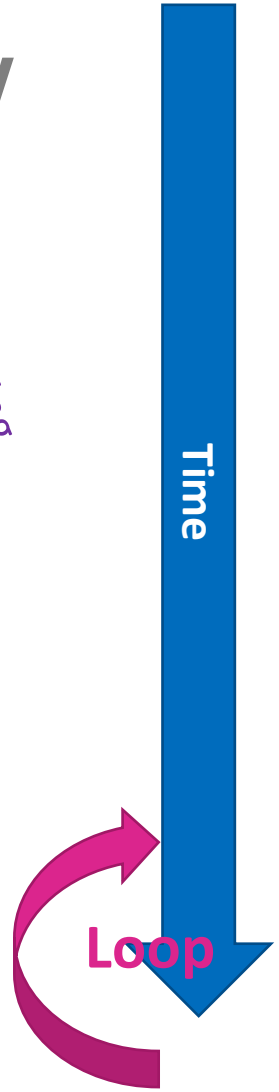
SDC 23

Pre-Copy Summary

Source SSD View

- Start Logging
- Copy Initial NS
  - LBA Mapped Status Query
  - Read Mapped Data
- Iterative Data Copy
  - Read data from parsed MQEs

Excess logging
Closes race conditions



Target SSD View

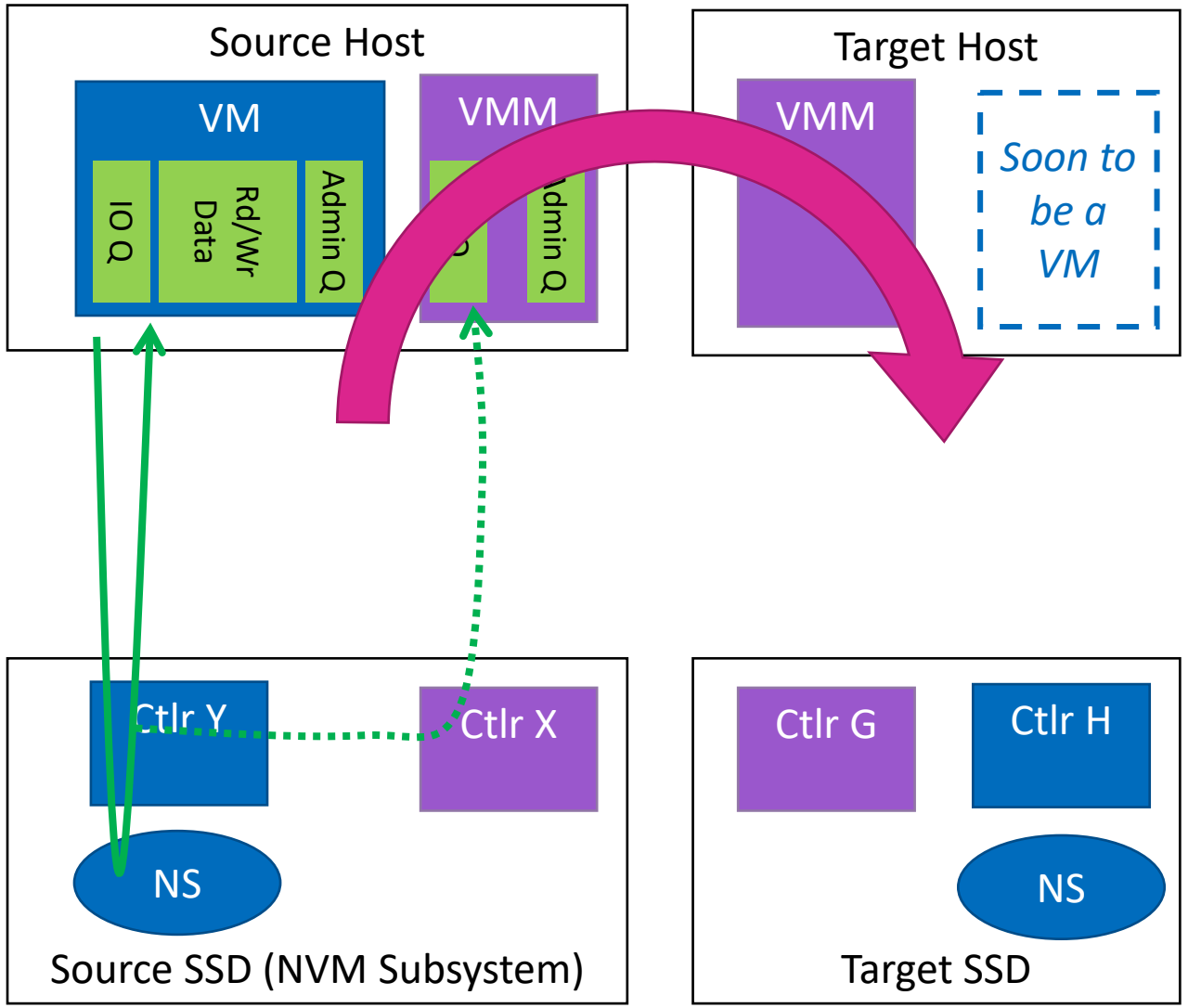
- Initialize Child
  - Initialize Child Ctlr
  - Create NS
- Copy Initial NS
  - Write Mapped Data
- Iterative Data Copy
  - Write data from parsed MQEs

SAMSUNG

SDC 23

Pre-Copy Phase: Iterative Data Copy

- Ongoing
  - VM has continued to Rd/Wr to Source NS
  - Source Primary Ctlr X has continued to log all appropriate activities to VMM
  - Copying from Source SSD to Target SSD takes time
- Source Drive View
  - Has experienced Reads from initial copy of Source NS to Target NS
  - Continues to experience more Reads from VMM parsing MQ logs
    - VMM is continuing to catch up to the VM's activity
  - Data is written to Target Child NS

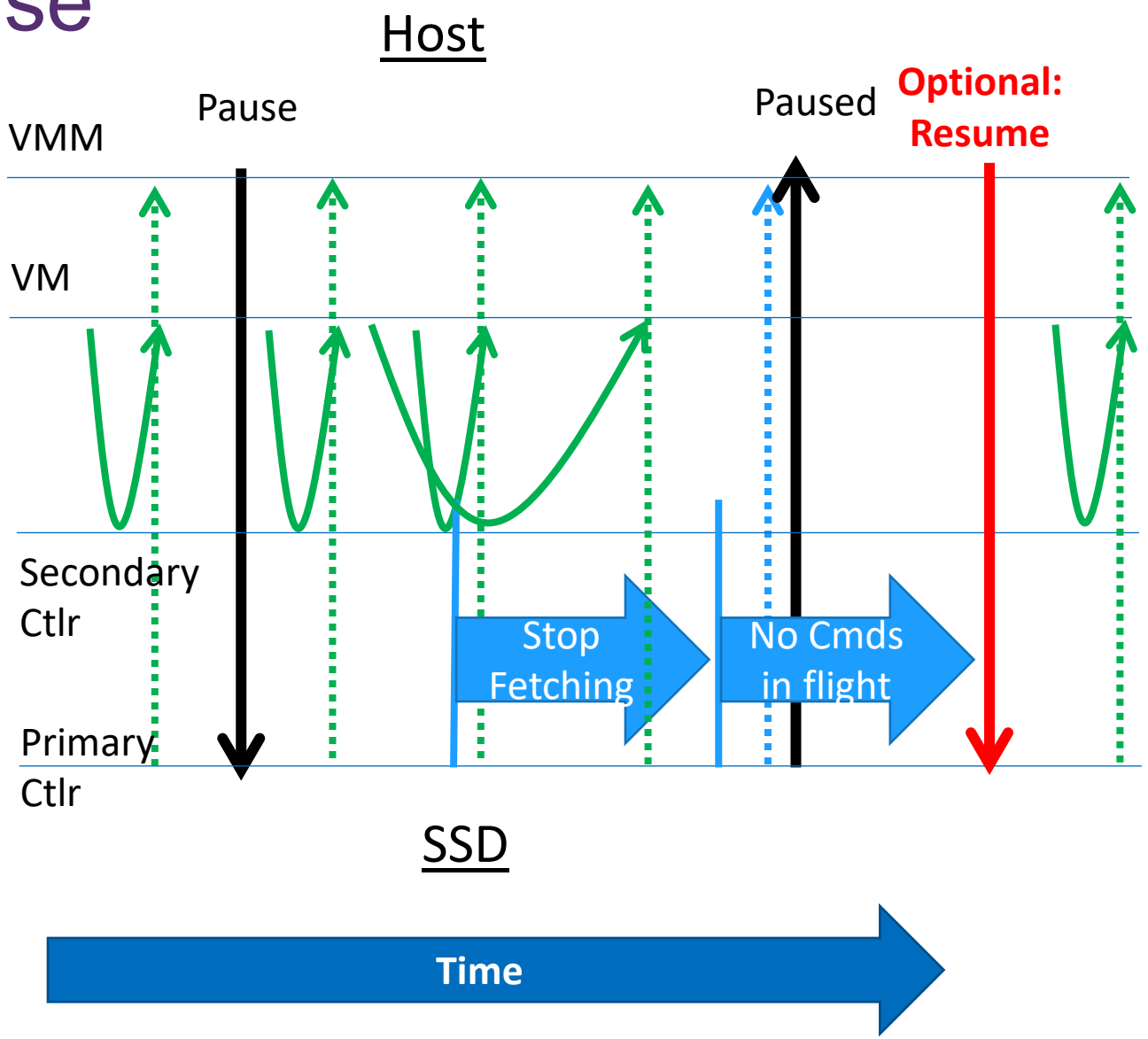


SAMSUNG

SDC 23

Stop and Copy Phase: Pause

- VMM decides to complete/execute the migration
  - VMM issues Pause Command to Primary Ctlr
- "Pause" Command Flow
  - Secondary controller stops fetching new commands
  - Secondary controller completes all commands in flight
    - Success vs Error are both acceptable
  - All CQEs are properly returned to VM
    - With any MQEs for logging
  - Primary Ctlr completes the Pause command to VMM
    - And may concurrently log this successful pause in the MQ
- Stopped status Summary
  - SQE/CQEs may be on the SQ/CQs of the VM
- Source SSD
  - Must be prepared for potential Resume Command
    - Perhaps due to a system error
    - Conceptually Resume/Start should behave the same on both Source and Target
    - Except: Source SSD would continue logging
  - If not resumed, expect Secondary Ctlr to be reset.
- VMM will
  - Parse all remaining MQEs
  - Copy any remaining data to Target Child NS

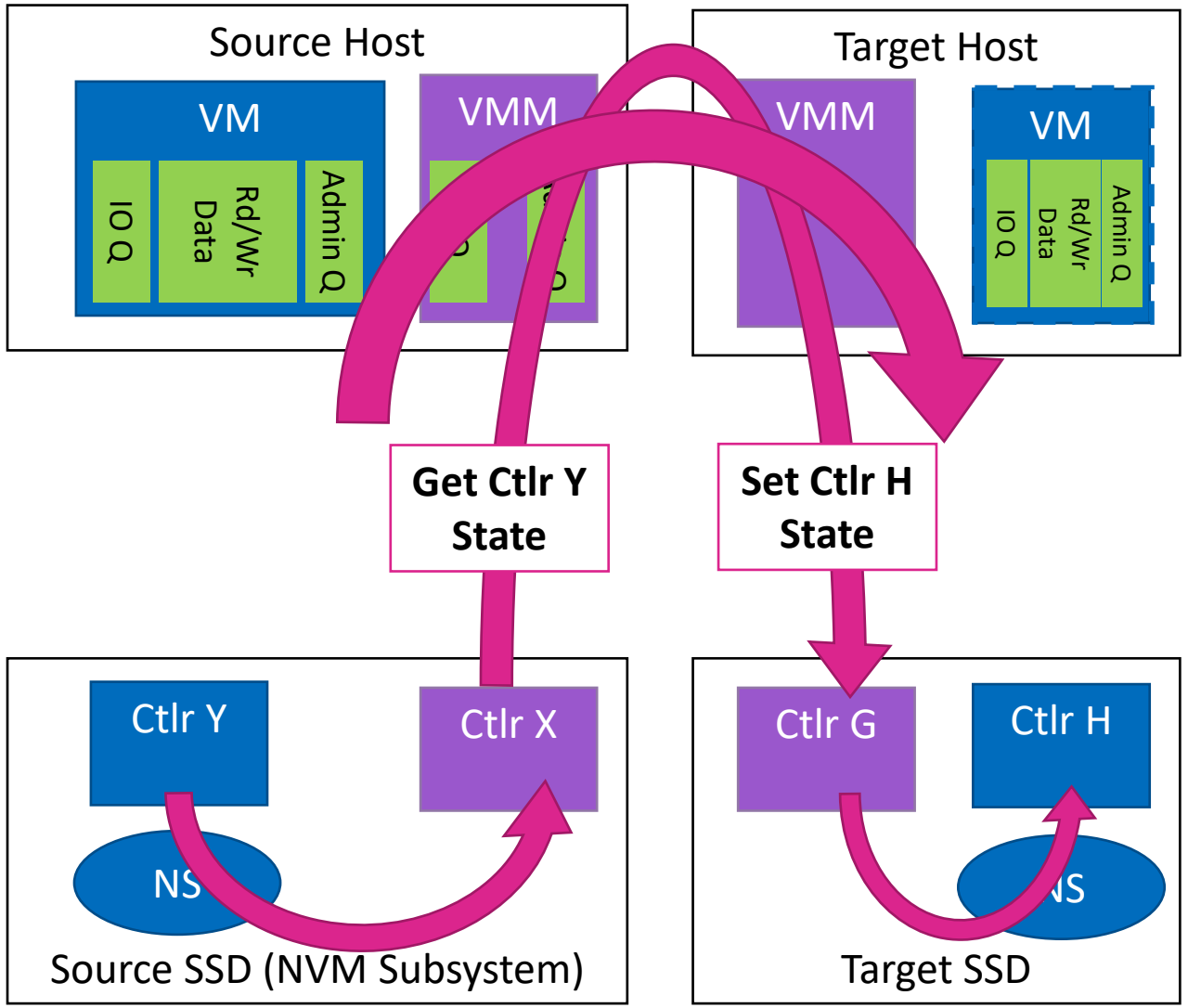


SAMSUNG

SDC 23

Post-Copy Phase: Copy Final Data and Migrate Controller State

- Final Data Copy Iterations from MQ Parsing
- Get/Set Controller State
  - Reads Ctlr Y out to the VMM
  - VMM Writes Ctlr H into the Target SSD
- VMM will migrate the VM
- From SSD's view
  - Same behavior:
    - Resume Ctlr Y sent to Ctlr X
    - Resume Ctlr H sent to Ctlr G
  - One difference: unlikely Ctlr G has enabled logging on Ctlr H
- Nominal NVMe Flows
  - Source VMM will clean up and reset Ctlr Y and NS



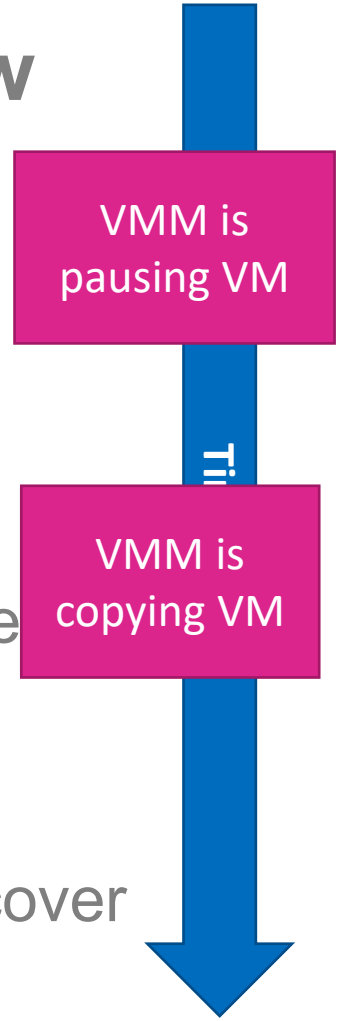
SAMSUNG

SDC 23

Finalizing Migration Summary

Source SSD View

- Stop-and-Copy
  - Pause
  - Read data tracked in MQ
- Post-Copy
  - Read Child Controller State
- Resume/Reset
  - Optional: SSD ready to recover from system error
  - Otherwise: VMM will reset Child Ctlr

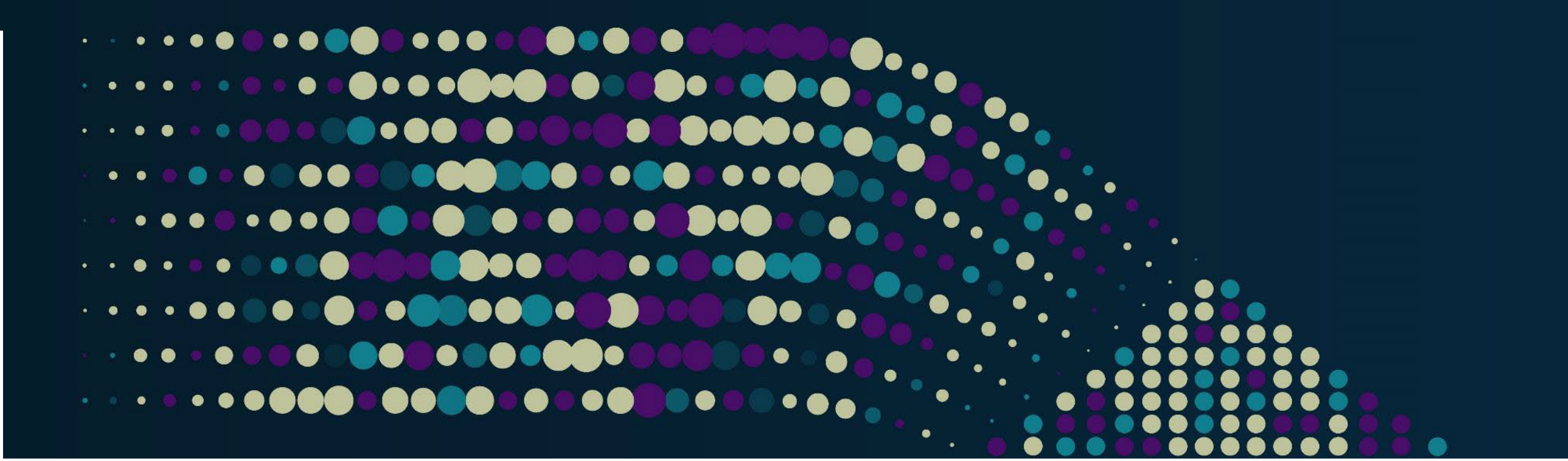


Target SSD View

- Stop-and-Copy
  - Write data tracked in MQ
- Post-Copy
  - Write Child Controller State
- Resume
  - Child Controller begins operating

SAMSUNG

SDC 23



Please take a moment to rate this session.

Your feedback is important to us.

SAMSUNG

SDC 23