SiliconMotion

Flash Memory Summit

Navigating SSD Tail Latency: Profiling, Analysis, and Optimization

Name: Guanying Wu (gwu@siliconmotion.com)

Title: Principal Engineer

Silicon Motion Technology Corp.

Legal Notice and Disclaimer

- The content of this document including, but not limited to, concepts, ideas, figures and architectures is furnished for informational use only, is subject to change without notice, and should not be construed as a commitment by Silicon Motion Inc. and its affiliates. Silicon Motion Inc. assumes no responsibility or liability for any errors or inaccuracies that may appear in the informational content contained in this document.
- Nothing in these materials is an offer to sell any of the components or devices referenced herein.
- Silicon Motion Inc. may have patents, patent applications, trademarks, copyrights, or other intellectual property rights covering subject matter in this document. Except as expressly provided in any written license agreement from Silicon Motion, Inc., the furnishing of this document does not give you any license to these patents, trademarks, copyrights, or other intellectual property.
-
- Silicon Motion, the Silicon Motion logo, MonTitan, the MonTitan logo are trademarks or registered trademarks of Silicon Motion Inc.

SiliconMotion

Problem Statement

Flash Memory Summit

- Solid-state drives (SSDs) face significant tail latency issues, stemming from factors such as garbage collection, wear leveling, NAND ECC, controller contention, firmware overhead, and task scheduling policies.

- In this presentation, we will discuss
  - ➢ the QoS Simulation model for effectively identifying the major contributors of long tail latency,
  - ➢ the optimization techniques for tuning SSD QoS.

SiliconMotion

Limitations of Existing Tools

Flash Memory Summit

Tool/Platform for QoS Analysis	Drawbacks
Behavioral model	Lack of timing for IO latency analysis
Cycle model	• Too much timing and too slow to simulate • Algorithm development as hard as FW development, almost like try-and-error
Analytical model	Mathematic Model used to estimate data-path performance and power, but challenging to deal with queuing network
ASIC	Easily get stuck in the complexity and details of HW/FW implementations and ineffectively identify the major factors contributing to the tail latency (Challenging to measure the system without causing much errors)

SMI Approach of QoS Tuning

SiliconMotion

SMI Proposal Summary

Flash Memory Summit

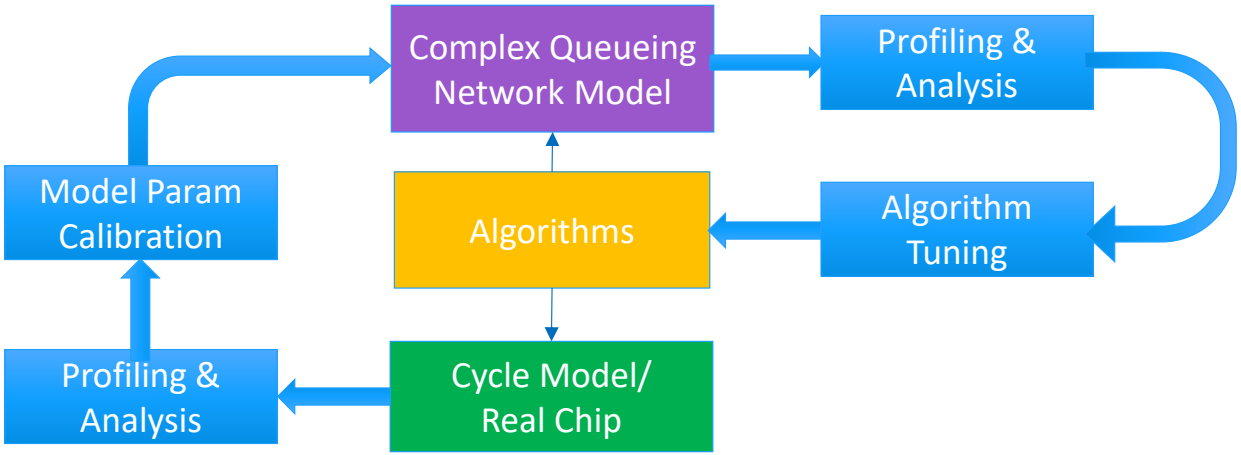
- The system within SSD is a complex queuing network
  - Buses, CPUs, DDRs, SRAMs, NANDs, Data-path processing engines, Control Path Accelerators, etc.
- Our model is based on queueing theory and real-time scheduling algorithms.
  - The model helps guide our profiling and analysis of this complex QoS problem.
  - SSD algorithms employed on the model are evaluated to check the effectiveness on QoS improvements.

Methodology of QoS Model

SiliconMotion

Methodology of QoS study

Flash Memory Summit



- Profiling
  - run the system various workloads/background ops
  - Collect queue statistics: utilization rate, service time, queue length
- Analysis
  - locate the hotspots (high utilization)
  - root cause the abnormal service time
- Algorithm Tuning
  - GC, WL, NAND Mgmt.
  - IO Traffic Scheduling
  - Power Shaping
  - Performance Shaping
- Model Calibration
  - Model Transaction Time Accuracy or Time Delay is calibrated based on Cycle-Accurate Model

Methodology of QoS study

- Modeling the system as a complex queuing network
- Profiling
  - run the system various workloads/background ops
  - collect statistics: utilization rate, service time, queue length
- Analysis
  - locate the hotspots
    - where utilization rate is the highest
  - root cause the abnormal
    - why a req is delayed so much?

SiliconMotion

Methodology of QoS study

Flash Memory Summit

- Algorithm Improvements
  - propose algorithms
    - FW centered algorithm development
      - FW determines what gets served first:
        - read vs write/erase
        - host vs GC
        - normal read vs retries
        - retries quick retries vs slow retries
    - key is to fine tune the parameters
- Verify the correctness
  - is it an algorithm that always halt with the expected behavior?
  - how
    - extensive testing
    - formal verification
- Validate the performance
  - how much improvement?

Design of QoS Model

Design Goals of the Model

- In need for a new tool that assists fast QoS algorithm development
- Simulation model
  - Queuing network model with timing realistically annotated
    - In between no-timing and too much timing
  - Features
    - Completeness
      - Includes all key components that impact timing
        - FW
          - IO processing
          - Background ops
            - GC
            - WL
            - NAND algorithms
            - System info maintenance
        - CPU
        - DDR
        - NAND
        - HW engines
- White box
  - Status of components and queues are fully visible and collectible
    - Find the bottleneck
  - The entire lifetime events of any request could be logged
    - Learn the latency distributions
    - Find the root cause of anomaly
- Easy error injection
- Fast dev
  - Modern C++ plus systemC
  - Quick algorithm exploration
- Formal verification

Design of the Model

- Written in modern C++ plus systemC
- Overall architecture
  - Timing annotated components interconnected via queues
  - Queues carry transactions, analogous to TLM non-blocking
- Profiling
  - Every host command is accompanied by a log of its state transitions and timestamp
  - How to help with queuing delay analysis?
    - When it is enqueued, record the number of customers ahead of it
  - Visualization utilities

QoS Optimization

SSD QoS as a Queuing System

- Focus on tail latency, not average latency
  - tail latency could be 10's times of average latency, even 100’s
- Tail latency comes from:
  - service time variability
  - queuing delay

Tail Latency: Service Time Variability

- Read retry:
  - source
    - there is always a rate of RR due to defects
    - but the major source is Vth misalignment
  - mitigation
    - find out the right Vth
      - at the cost of CPU/NAND bandwidth
- when RR happened: recover data quickly
  - RR sequence should be correctly sorted
    - need a complete profiling of the error rate and recovery rate
  - also need to be adaptive to
    - temp
    - PEC
    - retention
    - read disturb

SiliconMotion

Tail Latency: Queuing Delays

Flash Memory Summit

- Sources
  - utilization rate being too high
    - chance of contention goes high
    - intrinsic collision by randomness
  - head of line blocking
    - Prog/Erase
    - Read retries
  - fair utilization, but imbalanced
    - E.g., we have 2 parallel processing units but use only 1

SiliconMotion

Tail Latency: Queuing Delays

Flash Memory Summit

- Mitigation ideas
  - Priority
    - Foreground always prioritized
  - Preemption
    - Make background processes preemptible
      - Cancellation
        - Analogous to PE cancellation during PLP
      - Suspension
        - PE suspension
      - Slicing
        - Don't let a FW task occupying CPU for too long, break it into steps
    - Background starvation?
      - Reserve enough bandwidth (which background really needs) to make sure background processes can finish
      - Design scheduling policy so that background will never be too late for its purposes
        - Media scan should be able to run in time to get us the right Vth
  - Overprovisioning
    - Utilization is sacrificed
      - Overprovisioning resource
      - Make reservations for foreground jobs
        - Background processes can wait or be slower than foreground
        - Otherwise need to allow background to give up resources
    - Example, more CPU powers
- Trade life for latency
  - Do Garbage Collection (GC) during idle whenever possible
    - So that we don't have to do urgent GC
  - Make some bandwidth reservations for GC, even during foreground busy
  - Life will be shorter
    - But see how much head room we have
- Trade bandwidth for latency
  - Use some NAND bandwidth to do Vth tracking, so that Read Retry (RR) rate will be lower

SiliconMotion

Flash Memory Summit

Meet us at booth #315

Scan to learn more!