# A High Performance Reconfigurable Architecture for Flash File Systems

**Irfan MIR, Alistair McEWAN and Neil PERRINS**

**CPA 2012** **28th August 2012**

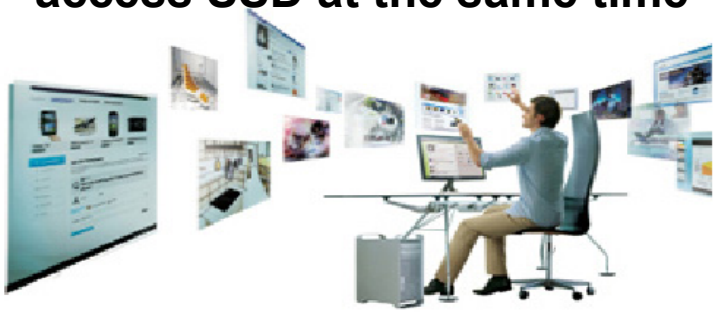
University of Leicester

Embedded Systems Research Group (ESRG),
Department of Engineering

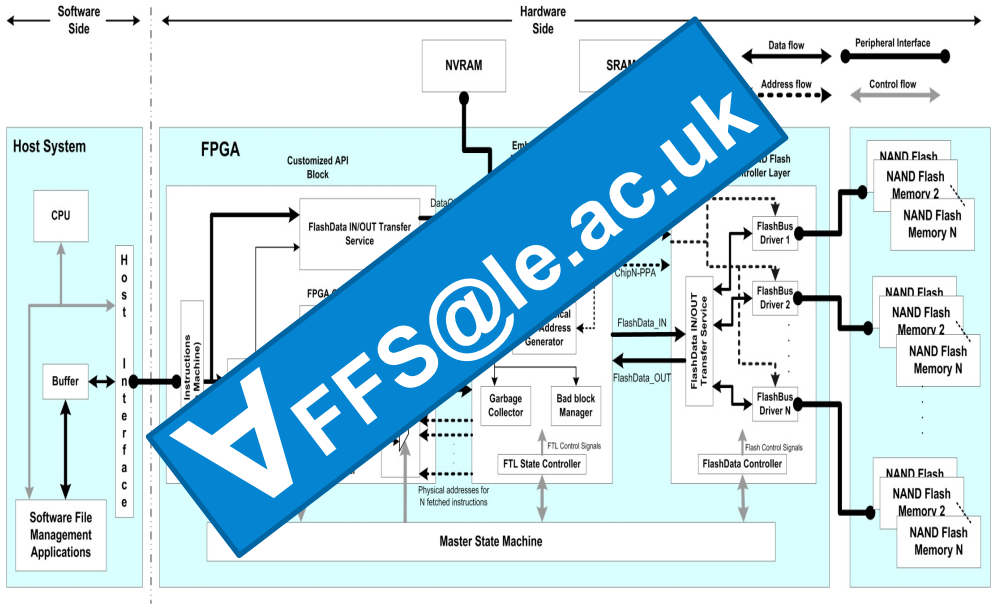
# Quick Overview

## SSD – Data Storage

## Multi-tasking: Multiple programs access SSD at the same time



## Flash Management Framework





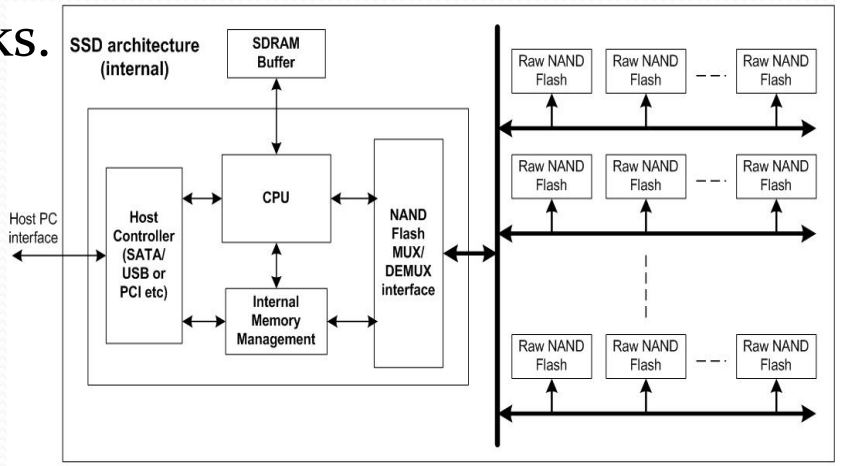

## Solution

# Outline

- Background
- Conventional Approach
- Exploit Concurrency
- Flash Management Framework
- Case study
- Performance Evaluation
- Conclusions and Future Work

# Background

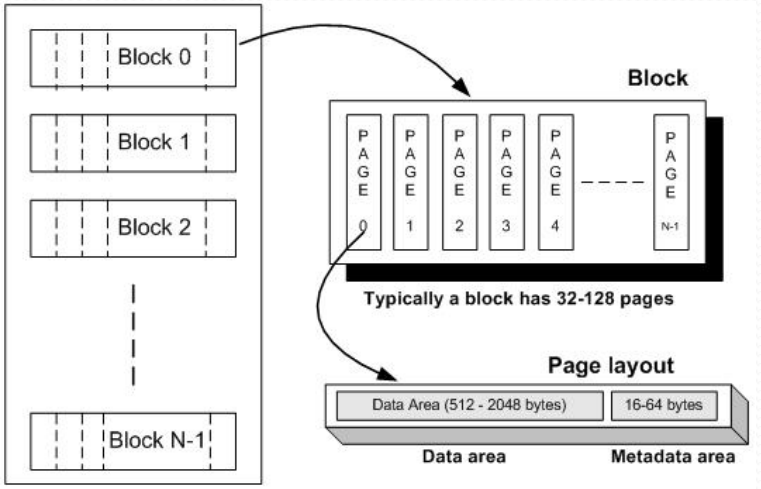
- Generic SSD architecture consists of a number of chips.
  - NAND flash consists of Erase Blocks.
  - Erase Block consists of pages.




- Advantages over magnetic disk.

- Performance degradation factors:
  - Erase-before-write operation
  - Garbage collection
  - Long fixed program/erase times
  - Multiple chips on a single flash bus

- Performance is a big challenge!



# Background

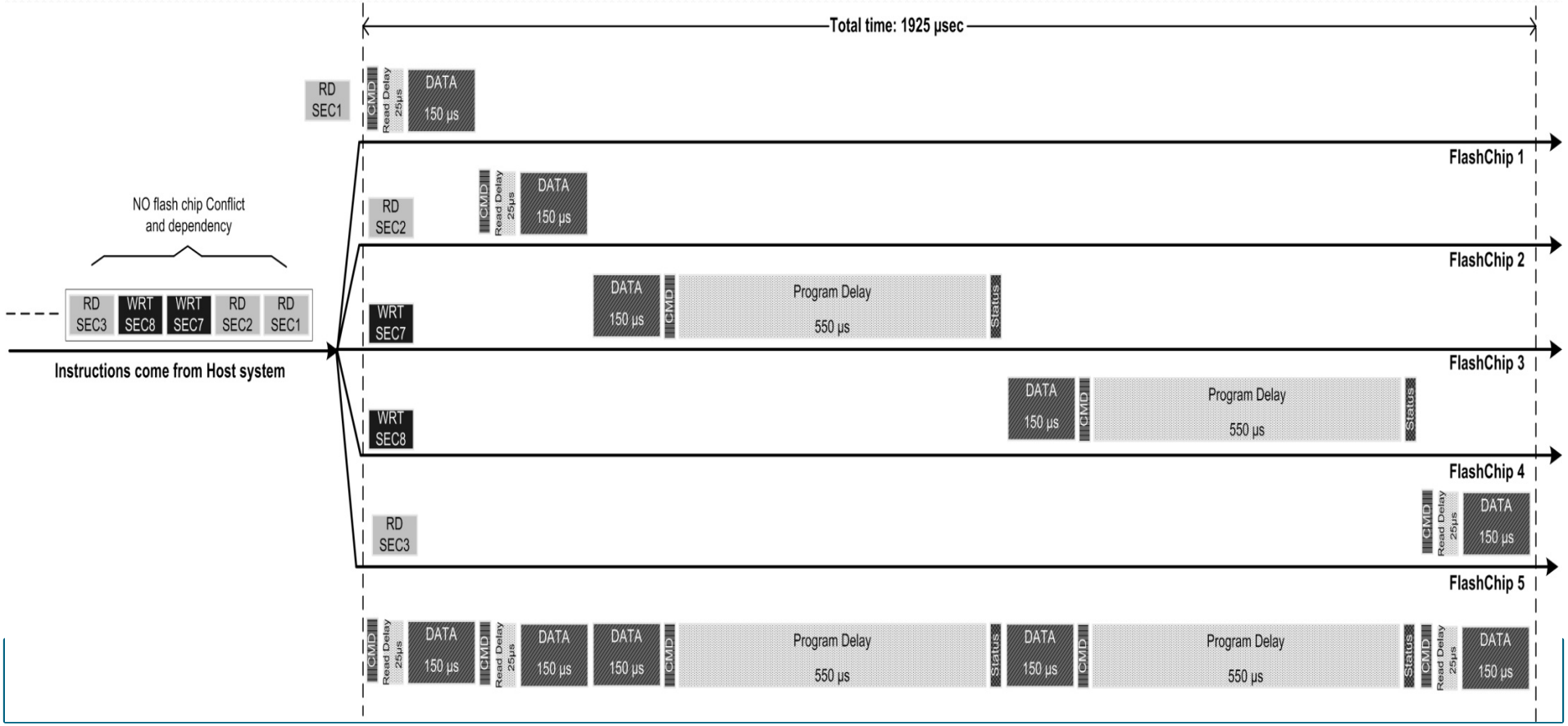
- Performance bottlenecks:
  - Multiple applications
  - Multiple access requests (write/read /erase operations).
- Bus architecture and Flash Translation Layer affect performance.
- FTL limitation factors:
  - Usually implemented in embedded software or in firmware.
  - Sequential in nature.
  - Limit operations in terms of throughput and response time.
- Commercial architectures are NOT revealed in public!
- **We have designed our own SSD architecture!**

# Outline

- Background
- Conventional Approach
- Exploit Concurrency
- Flash Management Framework
- Case study
- Performance Evaluation
- Conclusions and Future Work

# Conventional approach

- Performance bottlenecks:
  - Multiple operations on a single bus with multiple chips.
  - Flash operations are executed in the order in which they arrive!
  - Operations are executed in sequential manner.



# Contents

# Exploit concurrency!

- **Reconfigurable computing**
  - Parallel bus architecture.
  - Resulting in high throughput and fast response time.

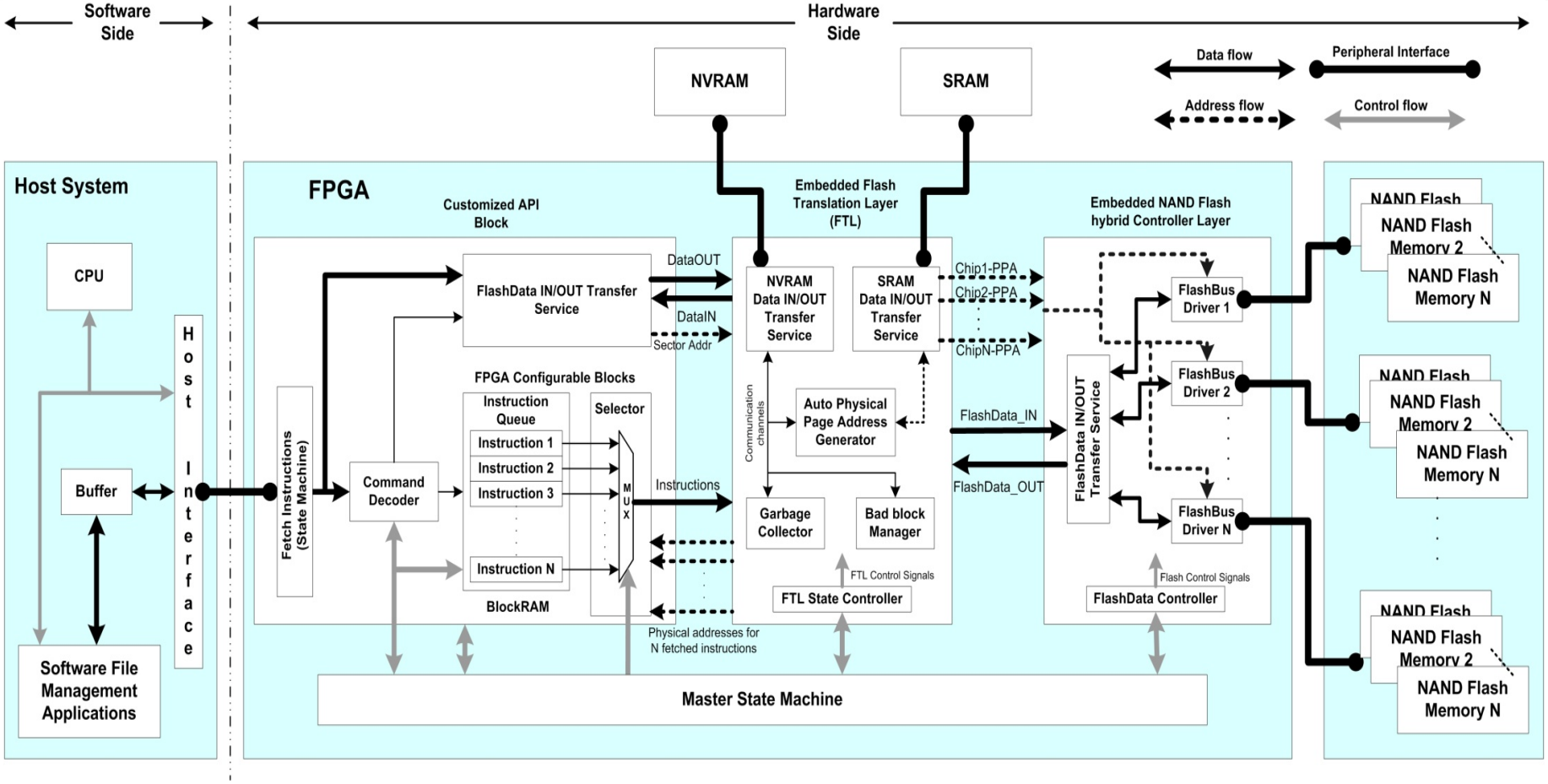
- **Concurrent programming language**
  - Our FFS is implemented  in Verilog HDL.
  - Thousands of concurrent hardware processes!

- We present a new **FPGA based flash management framework (FMF)** using reconfigurable computing, implemented in synthesizable Verilog.

# Contents

# Flash Management Framework (FMF)

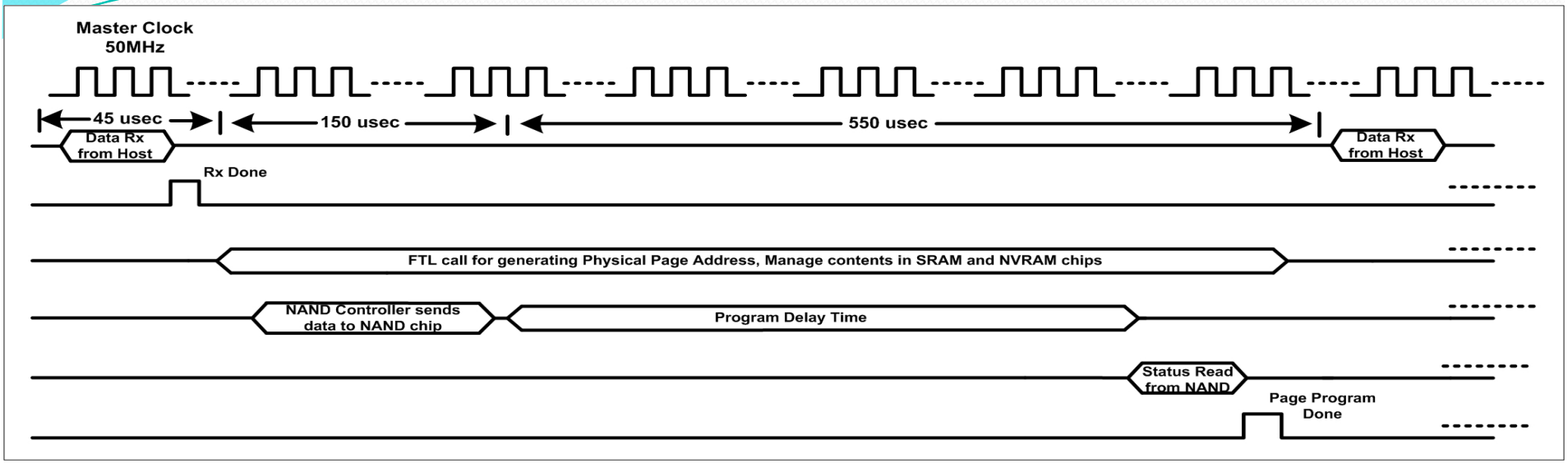
- **Major components:**
  - User data/instruction transfer protocol
  - Embedded flash translation layer
  - NAND flash hybrid controller layer



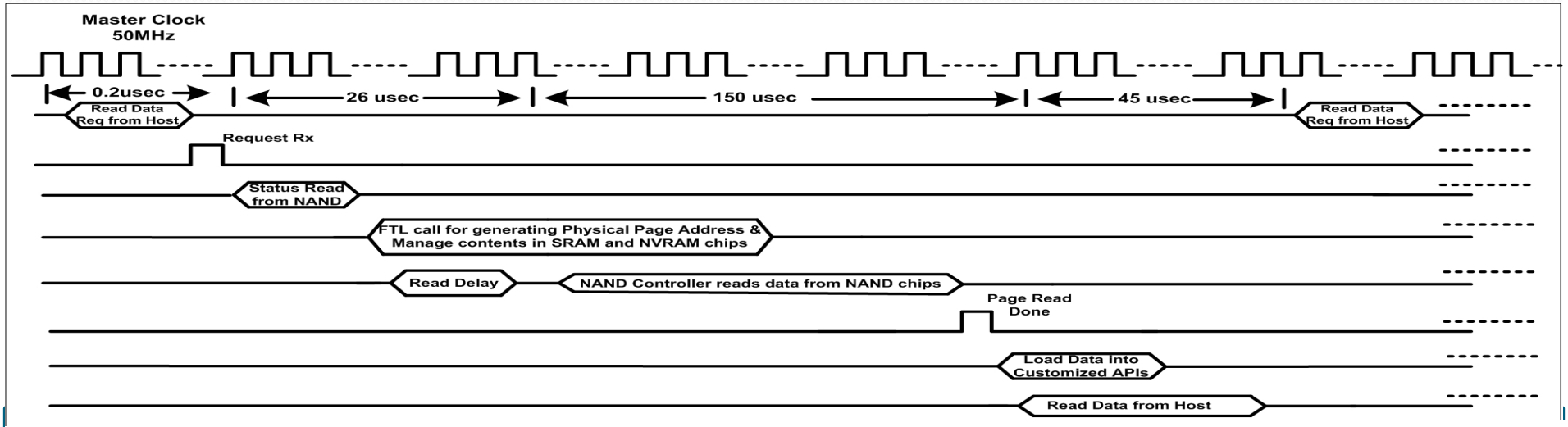
# Flash Management Framework (FMF)

- **Major features:**
  - Out-of-order execution exploiting multi-chip parallelism.
  - Command scheduling on ***X**Y*Z*** write/read/erase requests:
    - ***X*** - Flash buses
    - ***Y*** - Flash chips per bus
    - ***Z*** - Chip conflicts
  - Dynamic scheduling on the basis of **Erase-Write-Read (EWR)** command sequence.
  - Effectively handle chip conflicts and data dependency.
  - Dynamic chip assignment for write operations using auto physical page address generator mechanism.

# FMF - Flash operations




**Page write operation**

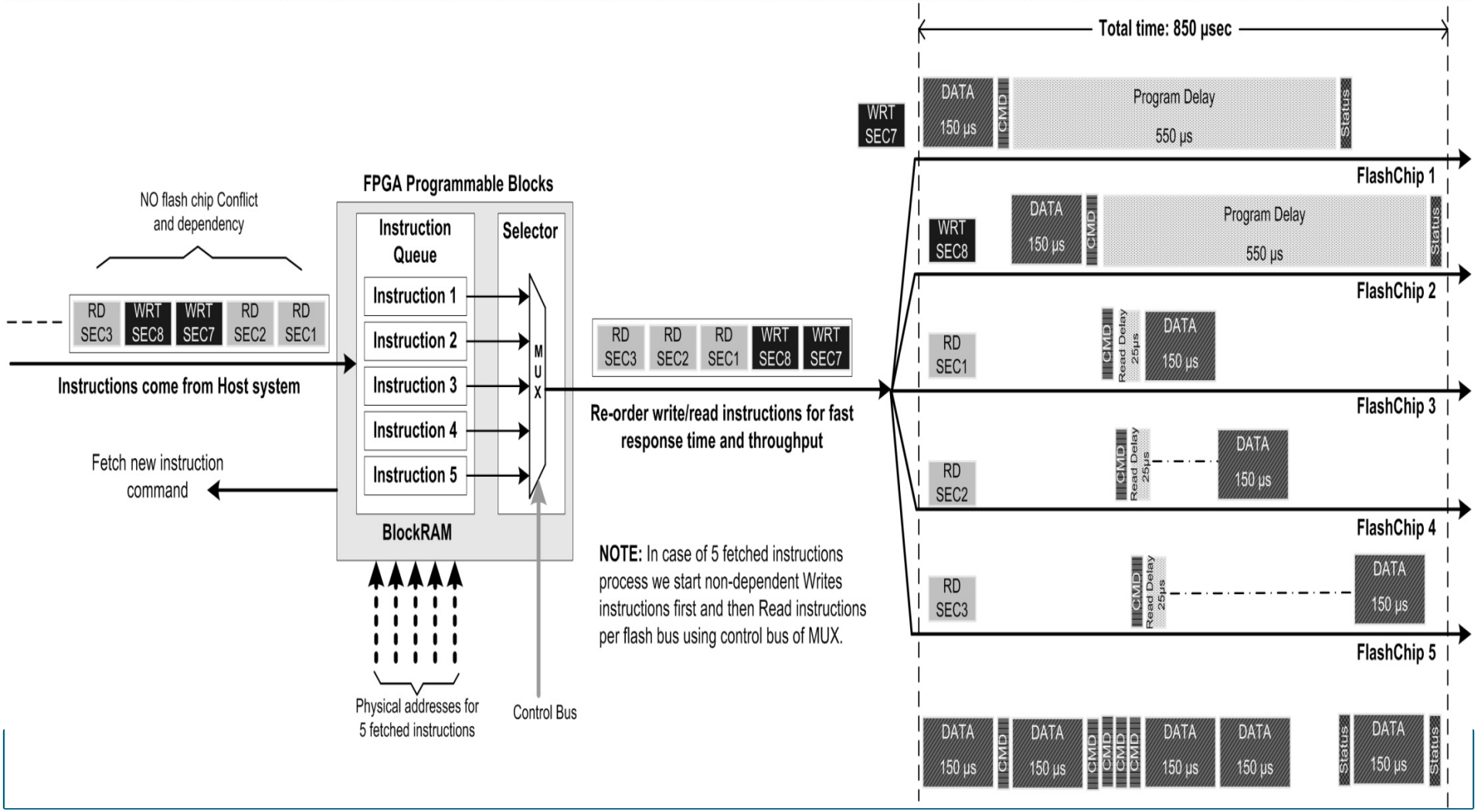



**Page read operation**

# Contents

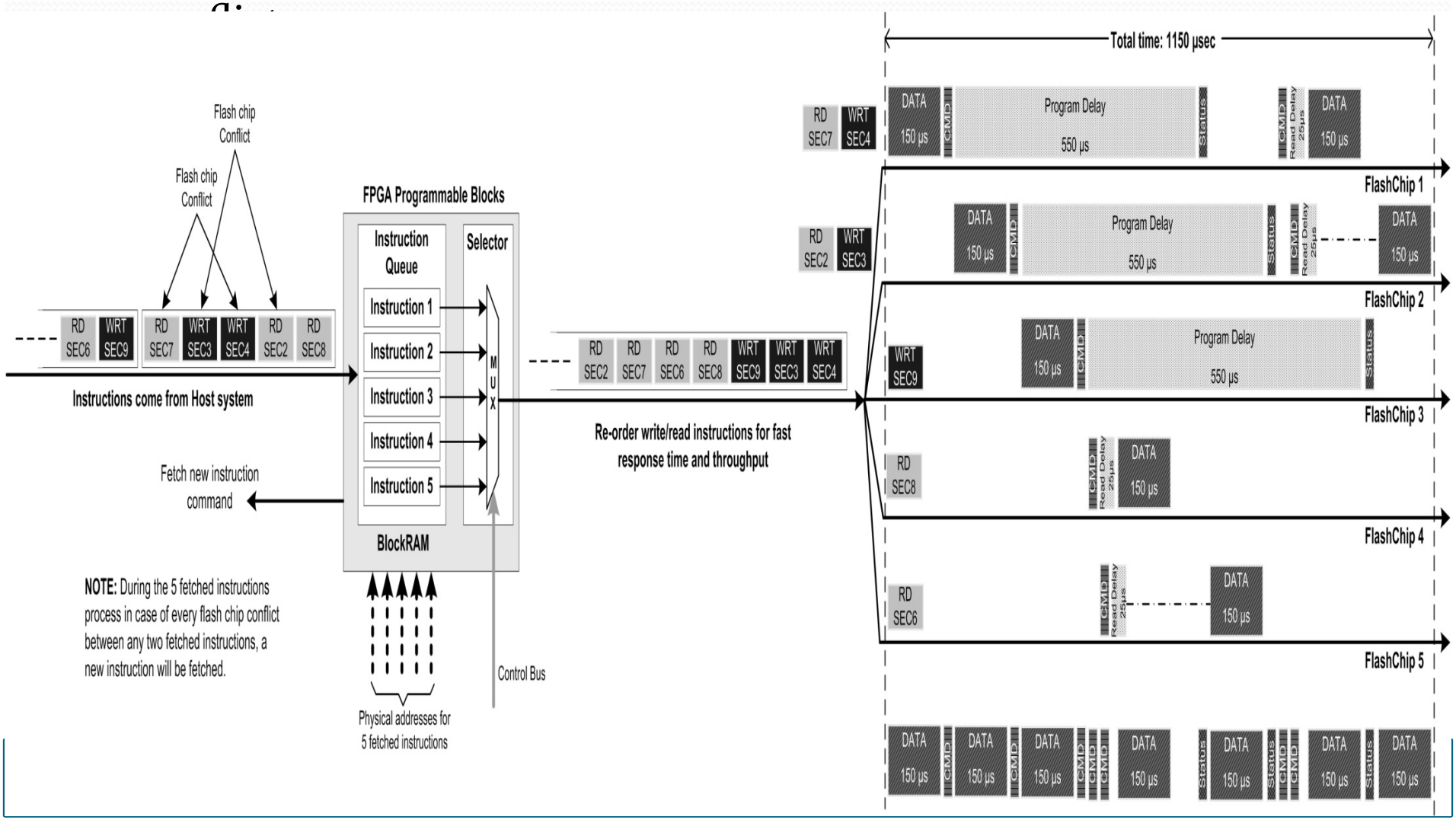
# Case study

- Scenario 1:
  - Multi sector write/read operations per flash bus with no conflicts.



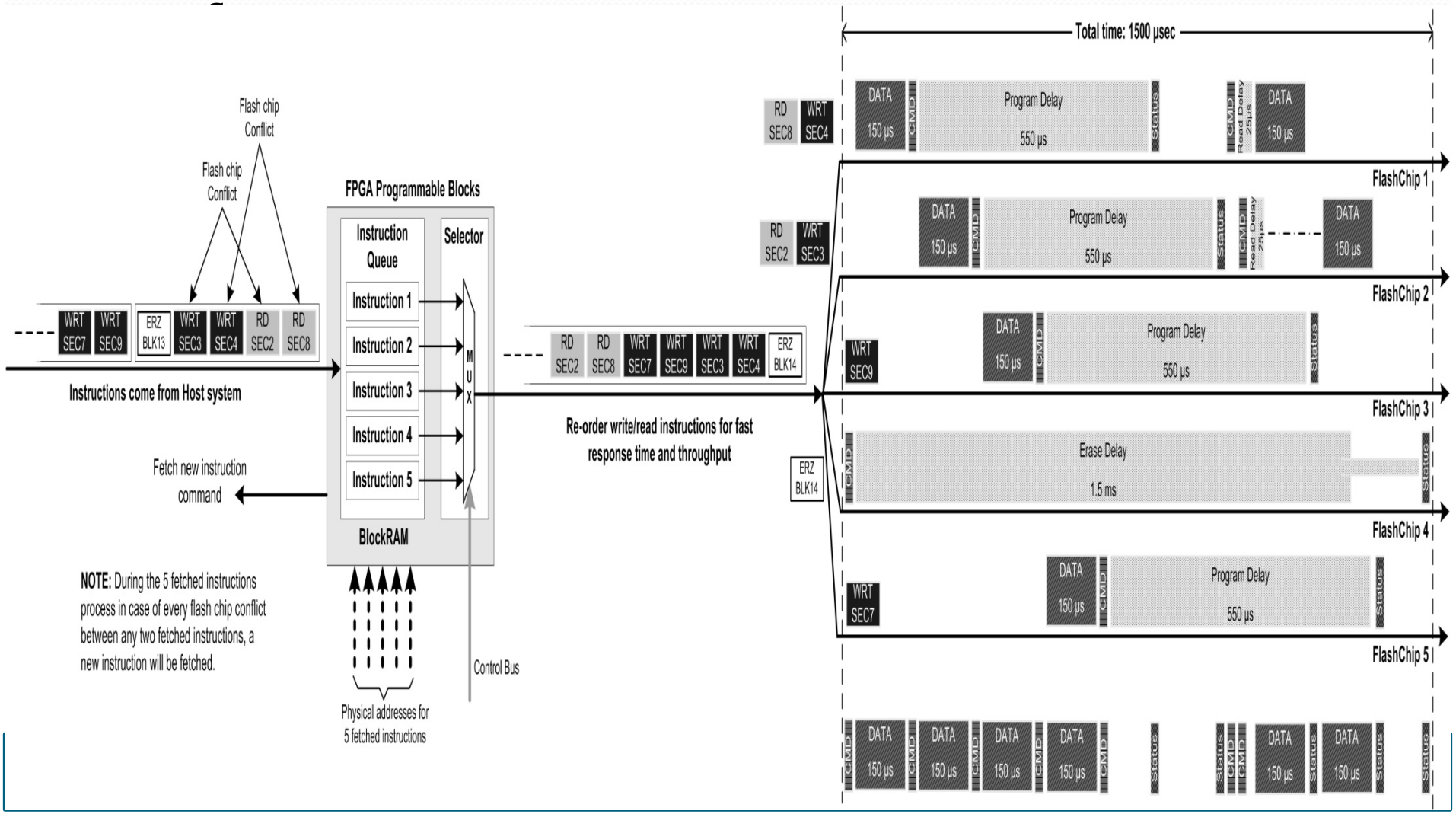
# Case study

- Scenario 2:
  - Multi sector write/read operations per flash bus with chip



# Case study

- Scenario 3:
  - Multi sector write/read/erase operations per bus with chip

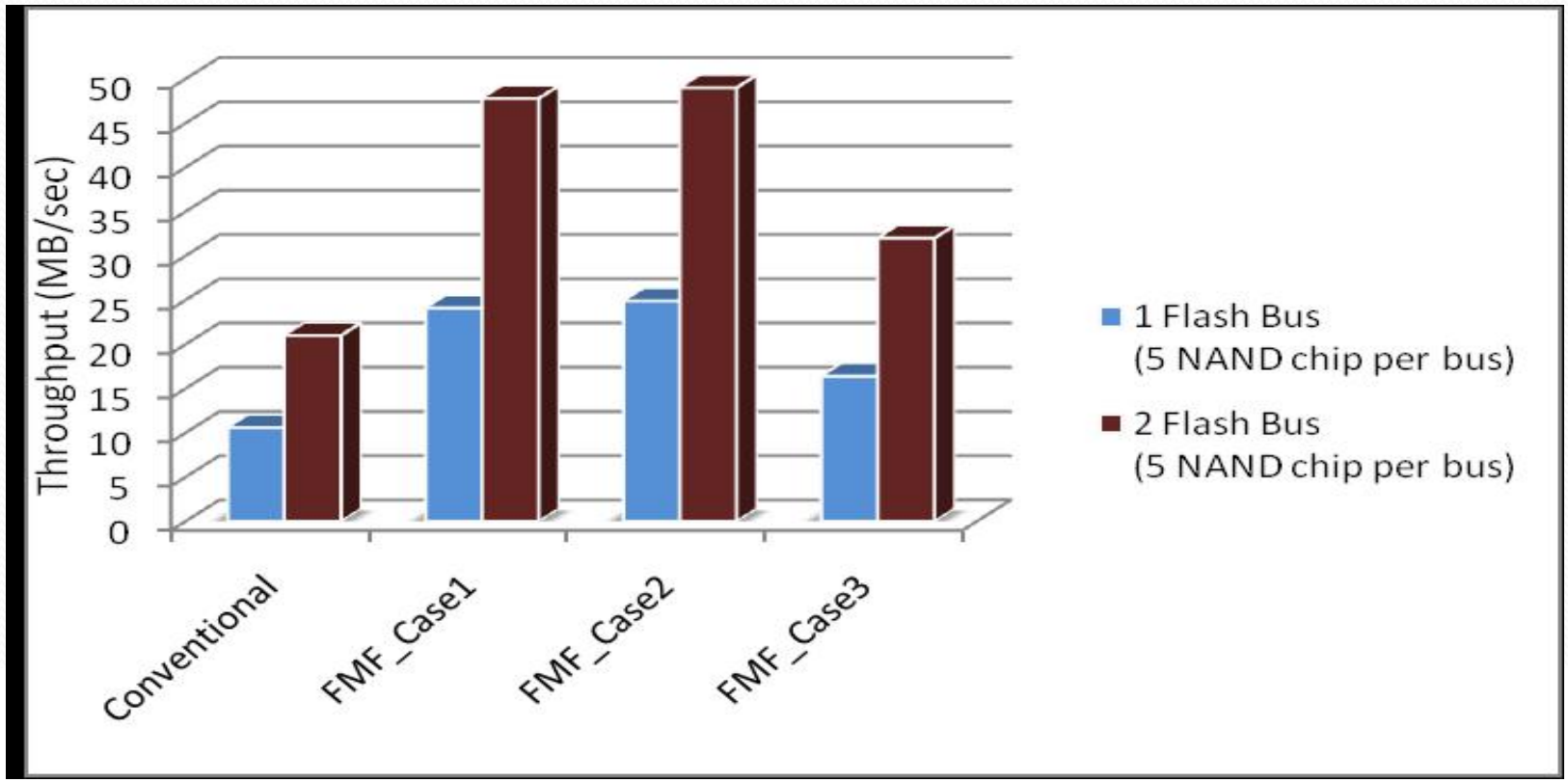


# Outline

- Background
- Conventional Approach
- Exploit Concurrency
- Flash Management Framework
- Case study
- **Performance Evaluation**
- Conclusions and Future Work

# Performance evaluation

- Parameters for experiments
  - Page and sector have same size (4KB).
  - Two 8-bit buses where each bus has 5 NAND chips.
  - Page read (25 µsec), Page program (550 µsec) and Block erase (1.5 msec).
  - Data transfer time of a flash page is 150 µsec.



# Outline

- Background
- Conventional Approach
- Exploit Concurrency
- Flash Management Framework
- Case study
- Performance Evaluation
- **Conclusions and Future Work**

# Conclusions and future work

- Conclusions
  - Our new flash management framework (FMF) is a viable way to develop a high-performance NAND flash-based file system.
  - It is mainly based on out-of-order execution technique with exploitation of multi-chip parallelism.
  - Our synthesizable Verilog implementation of FMF can be mapped onto any FPGA fabric.
  - Compared to conventional approach, our FMF provides a high throughput and reduced response time.

- Future work
  - We are building a parameterized test bench to evaluate our FMF in all possible scenarios under the synthetic and real workloads.