Toward a Unified Performance and Power Consumption NAND Flash Memory Model of Embedded and Solid State Secondary Storage Systems

Pierre Olivier*+, Jalil Boukhobza+, Eric Senn*
*+Univ. Europeenne de Bretagne, UMR6285, Lab-STICC
*Univ. Bretagne Sud, F56100 Lorient, France ; +Univ. Bretagne Occidentale, F29200 Brest, France
*{name.surname}@univ-ubs.fr, +{name.surname@univ-brest.fr}

Introduction

Contribution: a set of models to describe a complete flash storage subsystem.
- Covers the *wide specter of today's NAND applications* (embedded → mass storage)
- Designed to be implemented in a *simulation framework* (example : C++ API)

NAND Flash memory:
- Main secondary storage media in *embedded systems*
- Widely present in the *general computing* and *mass storage domains* (SSD)

Because of its benefits:
- Small size and weight
- I/O performance
- Power consumption
- Shock resistance

Models & Simulation tools:
- *Evaluate and compare* performance & power consumption
- *Estimate* these metrics in design phase
- *Prototype*, validate and test new systems

Existing tools [1-4] can be enhanced:
- Description details
- Consideration of power consumption
- Introduction of models rather than constants to describe behaviors
- Complete documentation

NAND Flash Memory

- EEPROM non-volatile memory (floating gate transistors) for data storage [5]
- Hierarchical architecture in NAND subsystem: Channels (A in schema below), Chips (B), Dies (C), Planes (D), Blocks (E), Pages (F)
- *Embedded storage* : generally one chip, *mass storage*: multi-chip, multi-channel

Legacy commands:
- Page read (G) and write (H)
- Block erase (I)

Advanced commands:
- Cache read and write (J)
- Copy back (K)
- Multi plane (L)
- Die interleaved (M)
- Multi chip (N)
- Multi channel (O)

(L, M, N, O: parallelism)

NAND constraints:
- *Erase-before-write*: a page must first be erased before being written
  - Out-of-place updates, old data invalidation
  - Garbage collection
- Wear of flash cells, R/W disturbance
- Target addresses of advanced commands

NAND management mechanisms:
- Raw chips: *Flash File Systems* (FFS, P in schema)
- SSD, USB flash drives, flash cards: *Flash Translation Layer* (FTL, Q)

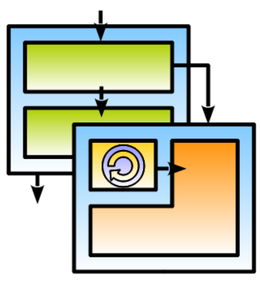
Modeling a NAND Flash Memory Storage Subsystem

We propose to describe a NAND flash based storage subsystem with a set of models: the **structural model**, the **functional model**, the **performance** and **power consumption** models.

Structural Model

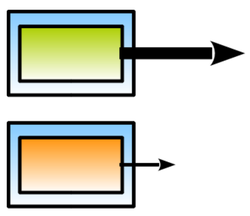
- **Description of the architecture and structural parameters of the NAND subsystem**
- Division into channels, chips, dies, planes, blocks & pages
- Num. of pages per block, page size, num. of channels, etc.
- Can be used to describe:
  - Simple embedded storage subsystems (one chip)
  - Complex multi-chip, multi-channel SSD storage subsystems

Functional Model



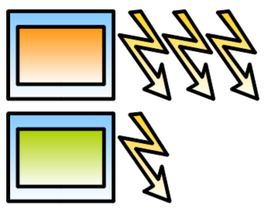
- **Functional behavior of the described storage subsystem**
- Supported commands:
  - Simple embedded storage subsystem: *legacy*
  - SSD / mass storage subsystem: *legacy* + *advanced*
- How to process these commands
- Defines and computes **flash events** according to an input command trace (R in schema). Ex : a page read operation
- Implements NAND constraints

Performance Model

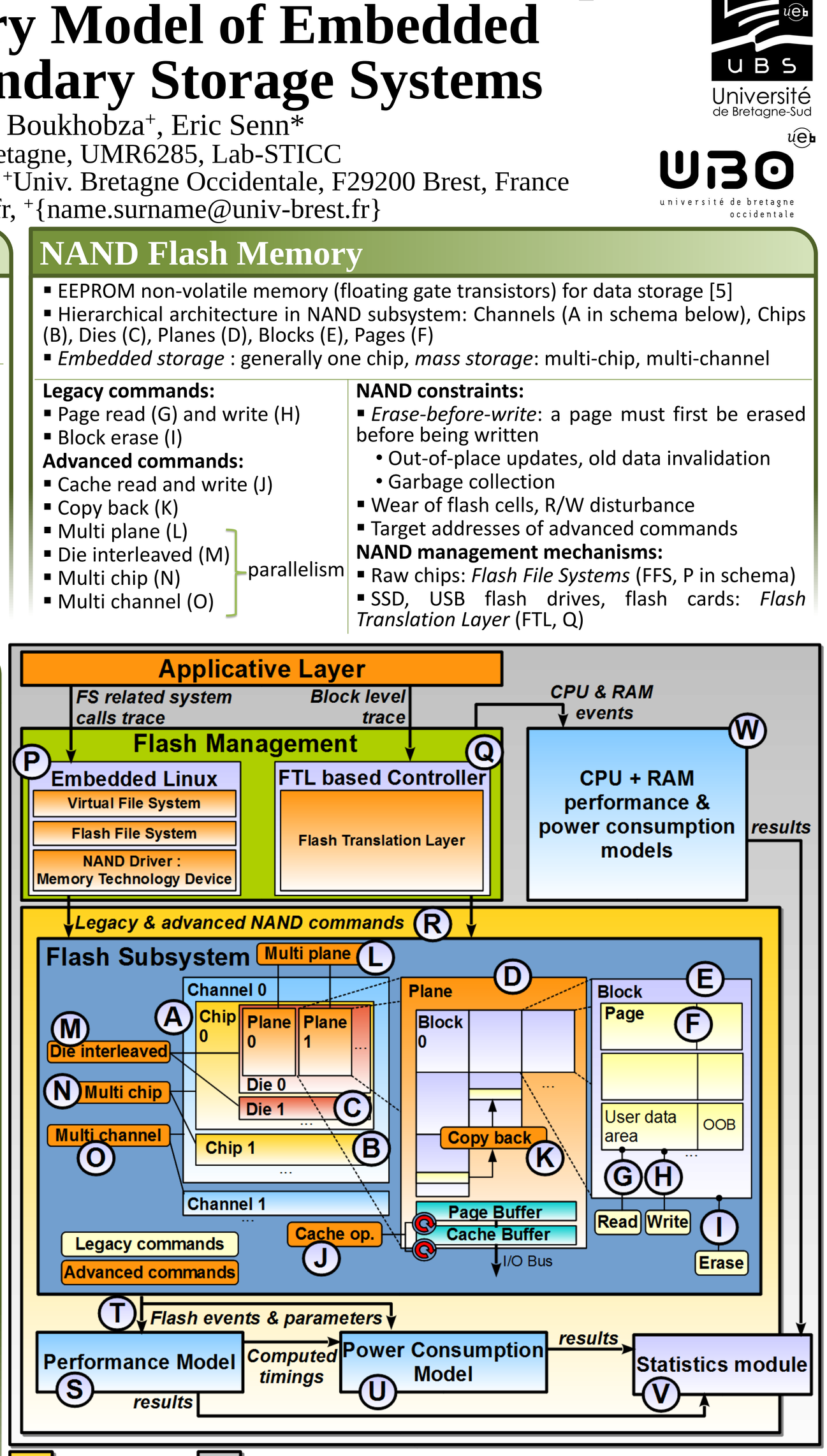


- **How to compute the execution time for the various flash events occurring in the system** (S in schema)
- A set of equations, one for each flash event defined by the functional model → to compute the execution time for these events
- Equations parameters are meaningful parameters for the event. Example: the address of the page read
- Flash events and parameters are passed by the functional model to the performance model (T in schema)

Power Consumption Model



- **How to compute power consumption for flash events** (U in schema)
- Similar to the performance model but targets power consumption
- In addition to events and parameters, takes computed timings as input from the performance model
  - For energy consumption computation



Models are designed to work together implemented in a flash storage subsystem simulation framework:
- Structural and functional models parameters are provided as inputs by the user
- Performance and power consumption meta-models: the user can input custom performance / power consumption models as sets of equations according to the described system
  - High level of abstraction providing accurate description of current and future flash subsystems designs
- Input trace: list of flash commands (R in schema) with parameters such as command type, address, and arrival time.

Global Project

This work is designed to be integrated in a simulation framework implementing models and simulating entire storage hierarchies for *embedded* and *SSD based* flash storage systems.
- Functional models for the control layer (P & Q on schema)
- Performance and power consumption models for hardware components CPU + RAM (W on schema)

Embedded Linux based flash storage system:
- Functional models for Linux storage hierarchy: VFS, FFS, NAND driver (P)
- Perf. and power cons. models for system CPU and RAM
- Input trace: FS related system calls

SSD based flash storage system:
- Functional models for the SSD controller (FTL): Allocation scheme, garbage collection, wear leveling
- Perf. and power cons. models for controller CPU and RAM (caches)
- Input trace: block level

Usage perspectives for such a framework are numerous:
- ***User level***, according to the inputs provided to the framework:
  - Performance / power consumption evaluation and estimation of existing flash management mechanisms and existing flash storage subsystems models
- ***Developer level***:
  - Bindings available to easily describe functional behavior for new complex flash management systems
  - For prototyping, testing and validation purposes

Implementation Example: C++ API

An implementation example in the form of a C++ API is proposed:
- Structural and functional models implemented as C++ classes and related methods
- Work in progress: performance and power consumption models integration
  - Abstract classes in the software, with function pointers members implementing equations for each flash event
- Statistics module (V on schema) collecting results and preparing simulation output
- Will be available online under GPL licensing

Conclusion and References

We propose a set of models describing a NAND flash based storage subsystem:
- Designed to be implemented in **simulation software** to evaluate and estimate performance / power consumption in existing and new systems
- Work included in a global project to model and simulate the **entire storage hierarchy** of embedded and solid state storage subsystems

[1] V. Mohan, S. Gurumurthi, and M. R. Stan, "Flashpower: A detailed power model for nand flash memory" , in *Design, Automation & Test in Europe Conference & Exhibition* (DATE), pp. 502–507. 2010.
[2] Y. Kim, B. Tauras, A. Gupta, D. Mihai, and N. B. Urgaonkar, "FlashSim: A Simulator for NAND Flash-based Solid-State Drives". 2009.
[3] M. Jung, E. H. Wilson, D. Donofrio, J. Shalf, and M. T. Kandemir, "NANDFlashSim: Intrinsic latency variation aware NAND flash memory system modeling and simulation at microarchitecture level", in *IEEE 28th Symposium on Mass Storage Systems and Technologies* (MSST), pp. 1 –12. 2012.
[4] Y. Hu, H. Jiang, D. Feng, L. Tian, H. Luo, and S. Zhang, "Performance impact and interplay of SSD parallelism through advanced commands, allocation strategy and data granularity" , in *Proceedings of the international conference on Supercomputing*, New York, NY, USA, pp. 96–107. 2011.
[5] J. Brewer and M. Gill, *Nonvolatile memory technologies with emphasis on flash: a comprehensive guide to understanding and using flash memory devices*, vol. 8. Wiley-IEEE Press. 2008.