# VST: A Virtual Stress Testing Framework for Discovering Bugs in SSD Flash-Translation Layers

Ren-Shuo Liu, **Yun-Sheng Chang**, Chih-Wen Hung

System and Storage Design Lab

Department of Electrical Engineering

**National Tsing Hua University, Taiwan**

NTHU

# Our Contributions — Virtual Stress Test (VST)

- Stress test SSD firmware without using real SSDs
- Enhance SSD firmware design flow
  - Fast stress tests surpassing SSDs' speed limitations
  - Scalable stress tests without a need of a large number of SSDs
  - Ease reproducing and investigating failed tests
- We apply VST to a real SSD project, OpenSSD, and VST helps us to discover seven new firmware bugs
  - A solid evidence of VST's bug-discovering effectiveness

# Outline

- Background
- Challenges and our contributions
- Virtual Stress Test design
- Evaluation
- Conclusions

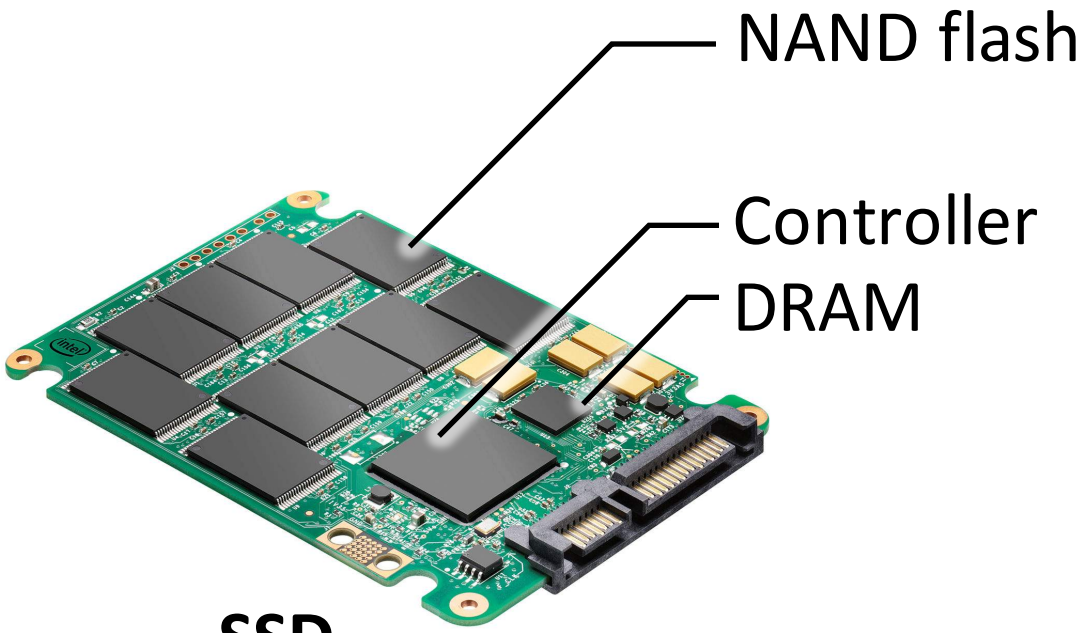
# NAND Flash-Based SSDs (Solid State Drives)

- SSD is an important storage technology now
  - Mobile, laptop, desktop, and server computers
- SSDs' advantages over hard disk drives
  - Superior access speed
  - Smaller form factors
  - Lower power consumption
  - Complete shock resistance

HDD

SSD

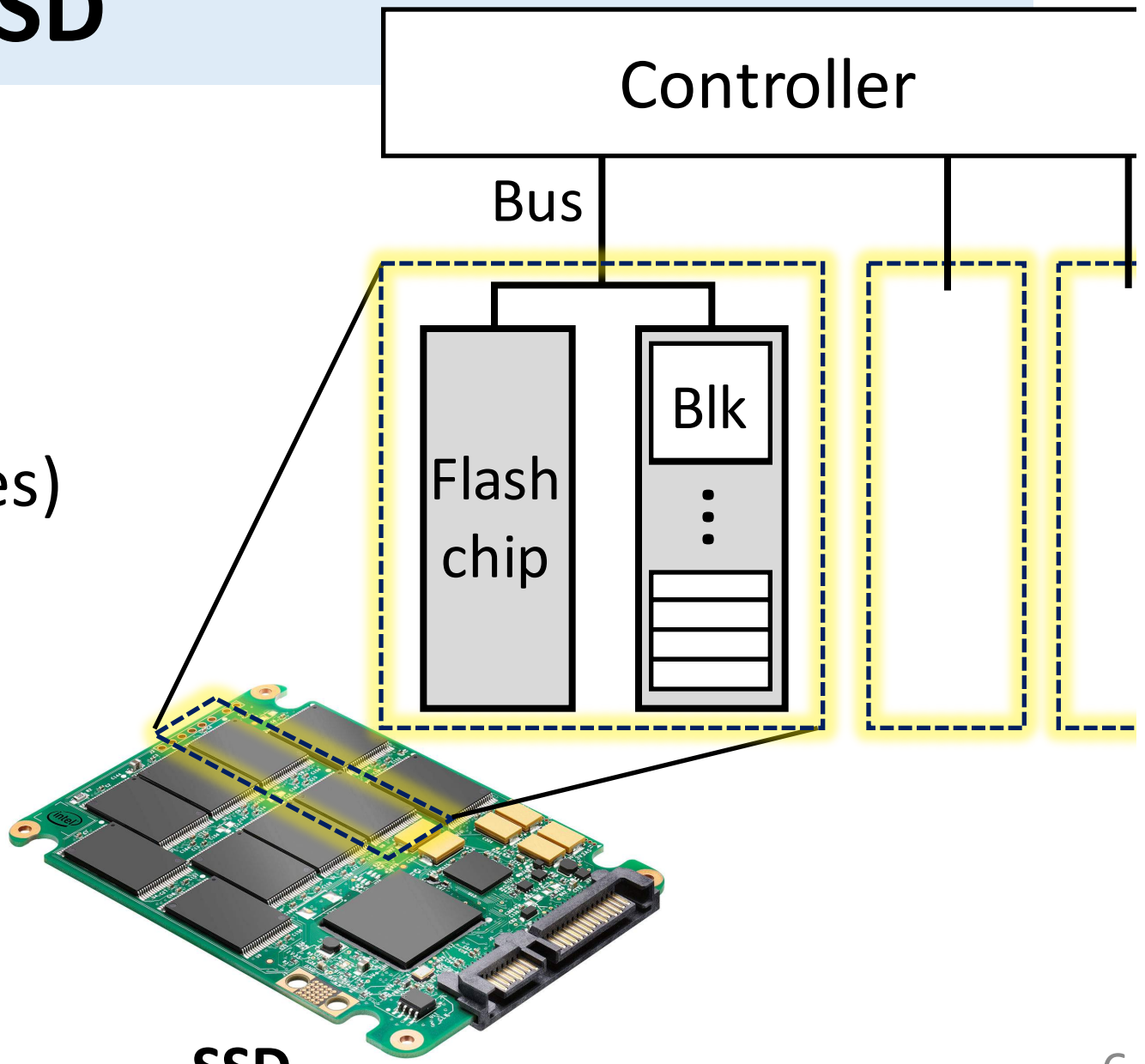
# NAND Flash-Based SSDs (Solid State Drives)




**SSD**

# Organization of an SSD

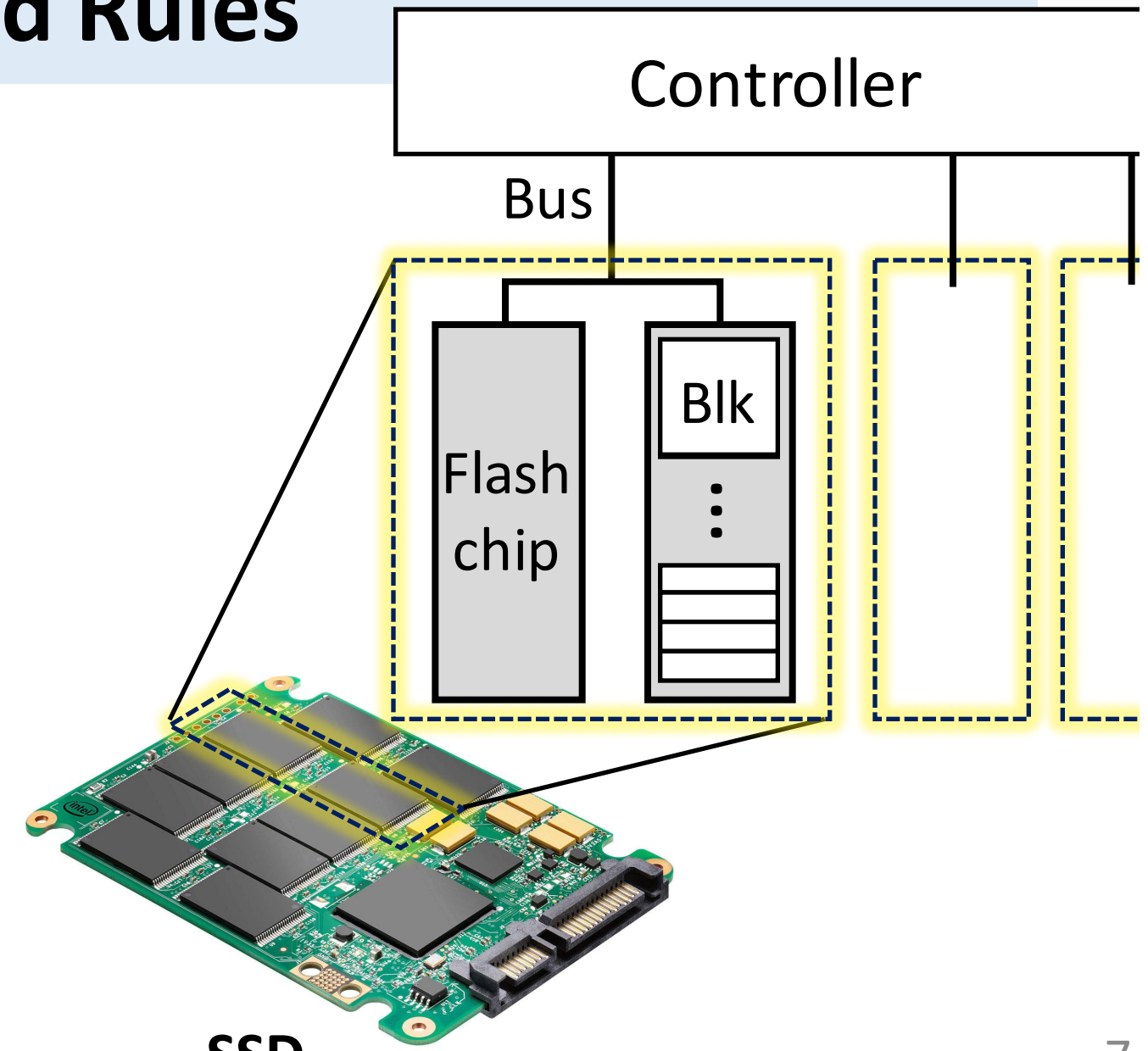
- Controller (1~8 buses)
- Bus (2－8 flash chips)
- Chip (~4096 flash blocks)
- Block (128－512 flash pages)
- Page (4－32KB)




SSD

# Flash Operations and Rules

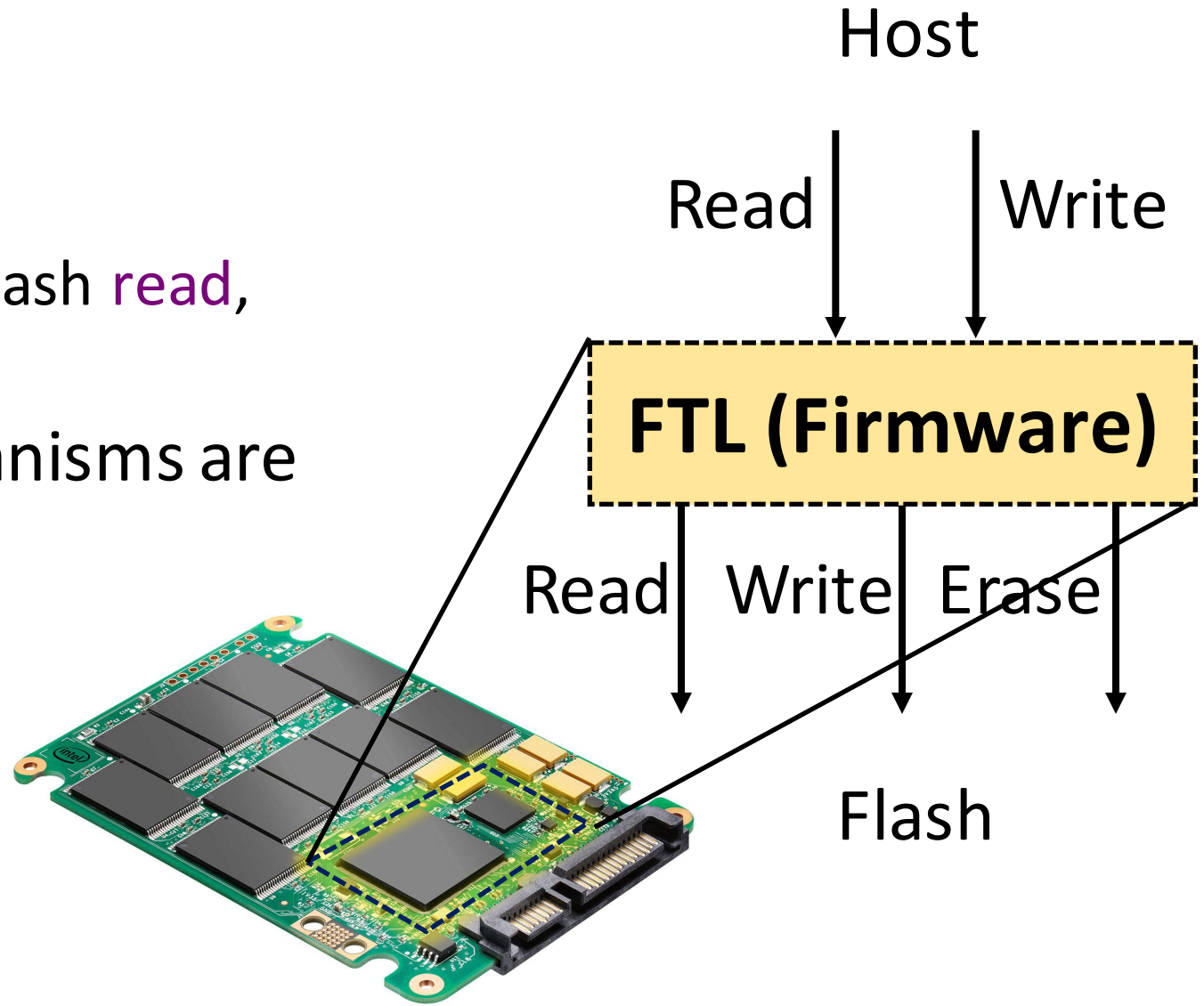
- Basic flash operations
  - Reading a page
  - Writing a page
  - Erasing a block (e.g., 256 pages)
- Basic rules
  - Directly overwriting a page is prohibited
  - Writing non-sequential pages (in a flash block) is prohibited




**SSD**

# Flash Translation Layer (FTL)

- The firmware in every SSD
- Basic FTL functions
  - Translate host requests to flash read, write, and erase operations
- Many advanced FTL mechanisms are proposed to improve FTL
  - Wear leveling
  - Hot-cold data separation
  - Dynamic write allocation



# Outline

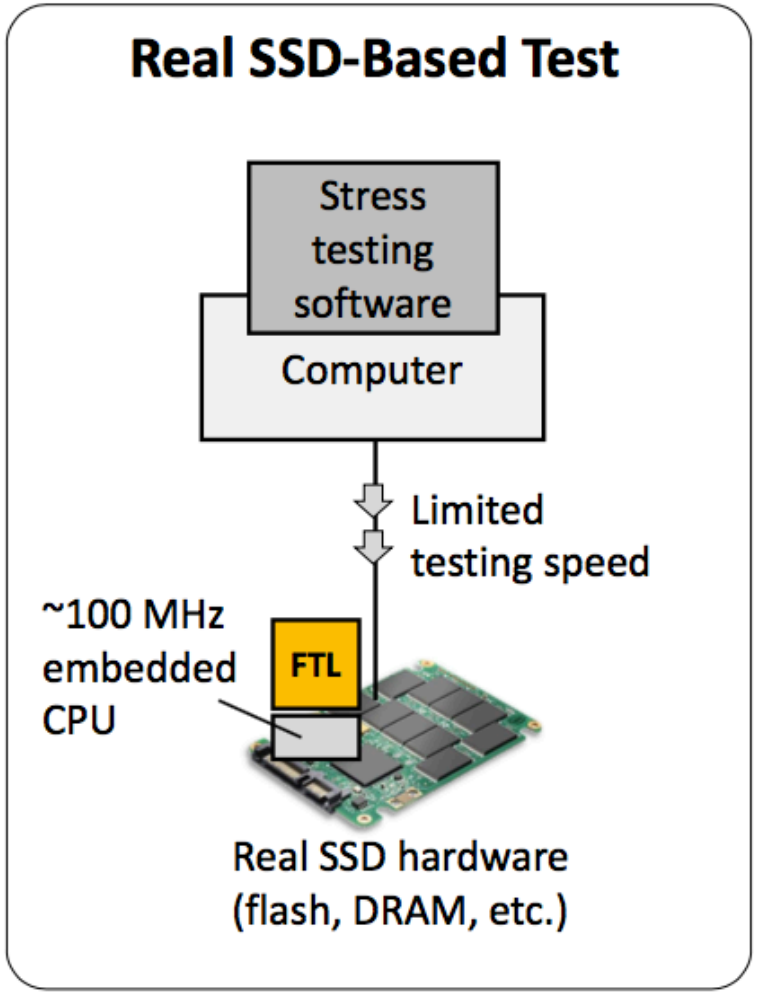
- Introduction and background
- Challenges and our contributions
- Virtual Stress Test design
- Evaluation
- Conclusions

# Challenges

- The pursuit of advanced FTL mechanisms increase FTL complexity
- More complex firmware are more prone to have bugs
- FTL bugs can lead to unacceptable and unrecoverable errors such as data and capacity losses

# Challenges

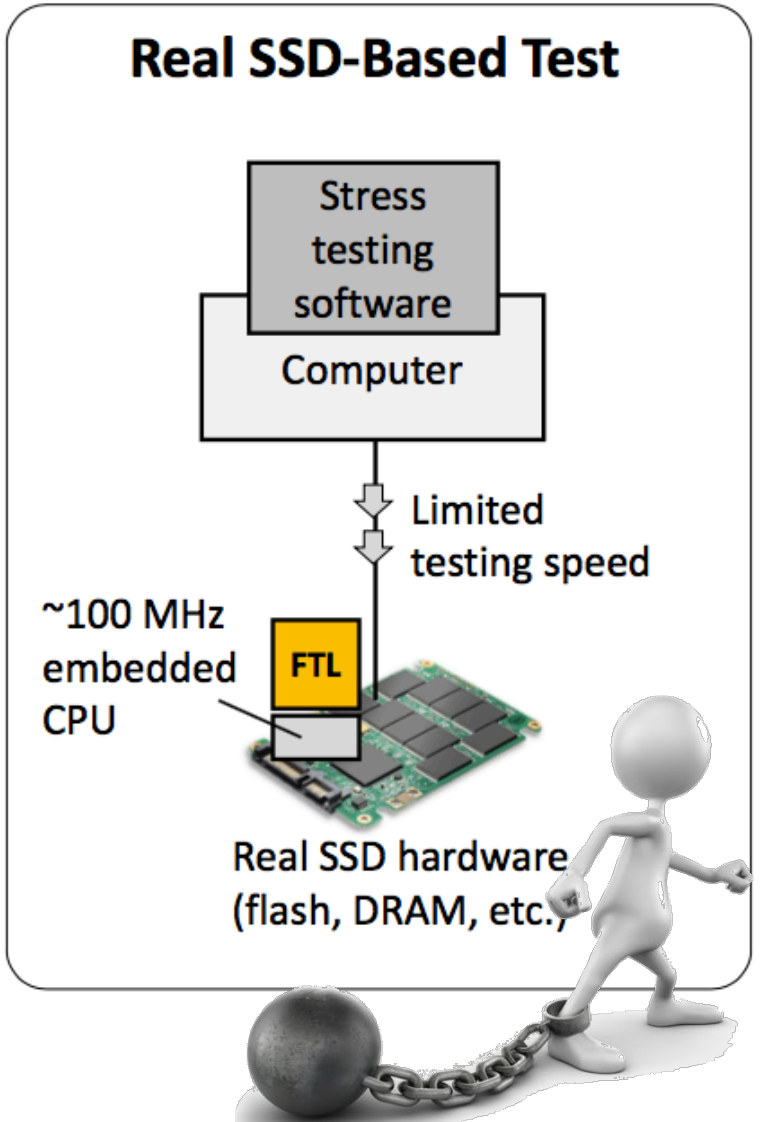
- Real SSD-based stress test is the common practice to discover FTL bugs
  - Generates intensive read and write requests to an SSD
  - FTL is probably buggy if abnormal behaviors present, such as
    - Request timeout
    - SSD disconnection
    - Data comparison mismatch



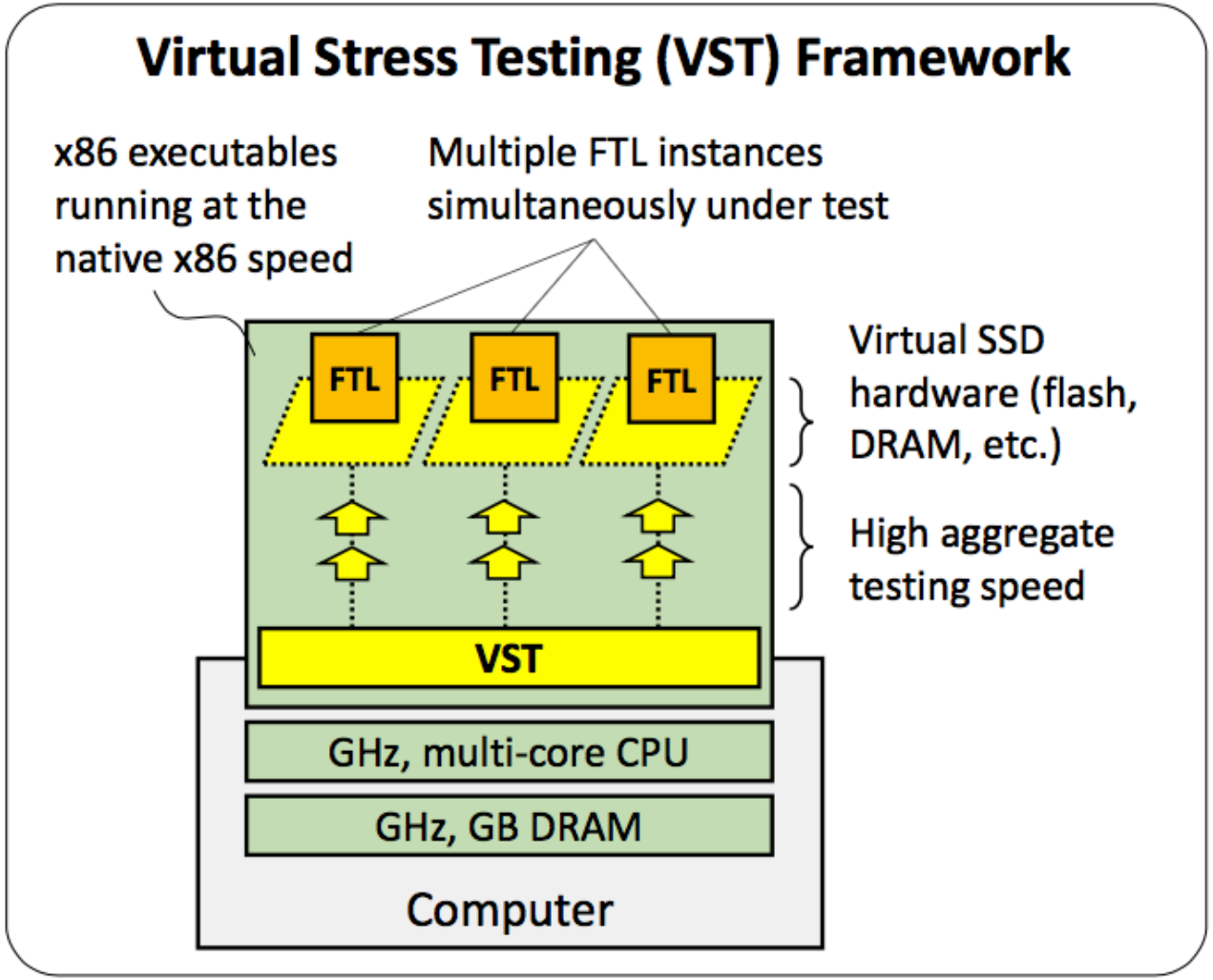
# Challenges

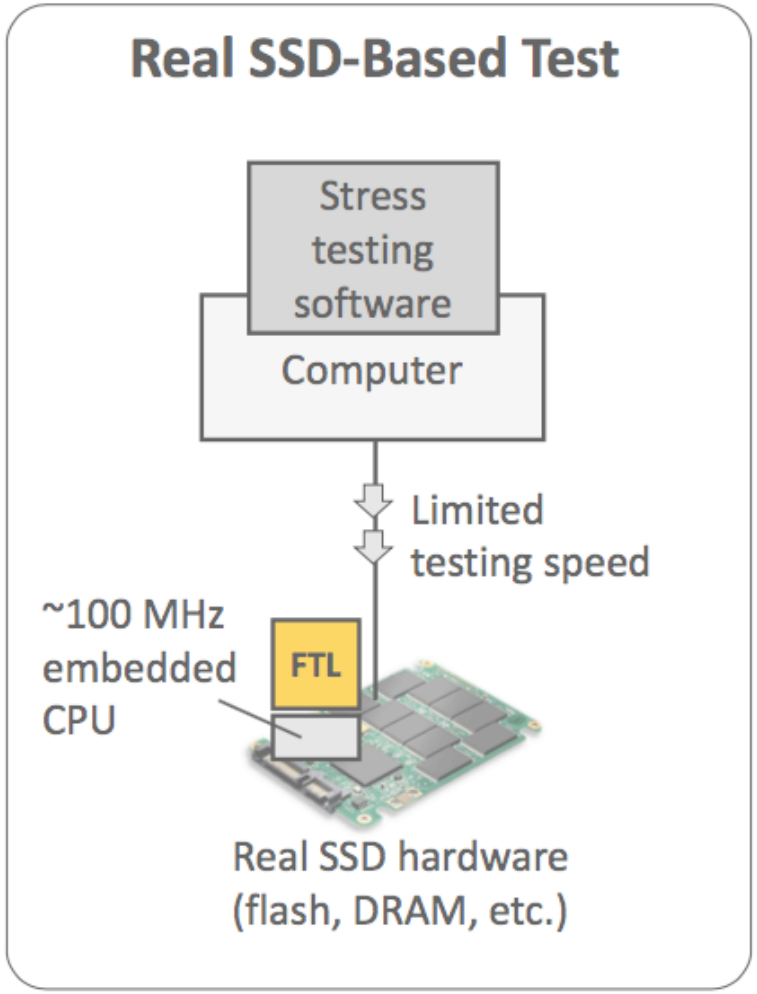
- Real SSD-based stress test is and will still be necessary

- However, it has drawbacks
  - Speed is limited by SSD hardware (e.g., flash memory)
  - Scalability is limited by the number of SSDs prepared
  - Reproducing a failed test can be difficult
  - Investigating a failed test needs expensive equipment (e.g,. a JTAG debugger)



# Our Solution — Virtual Stress Test (VST)



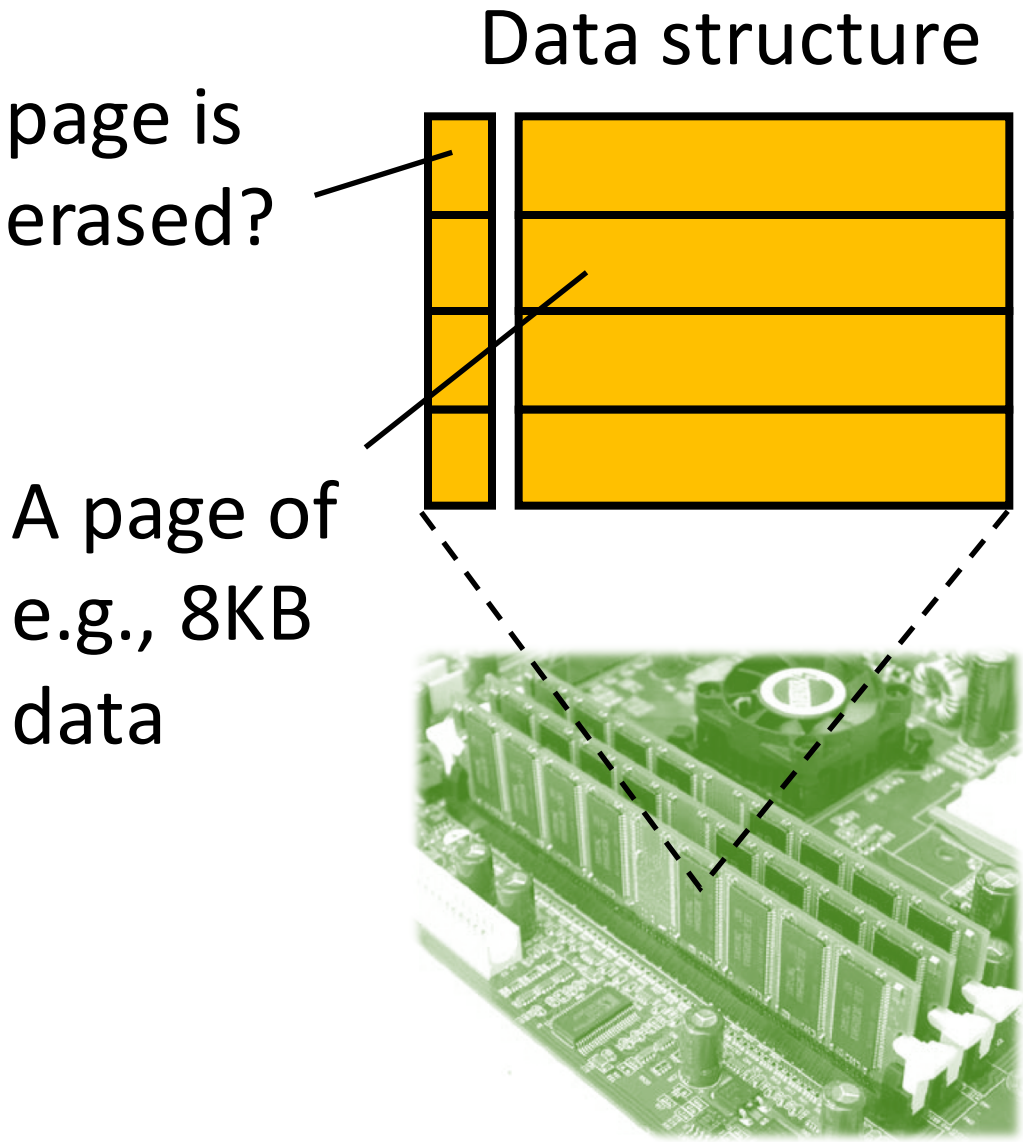
# VST vs. Traditional Stress Tests

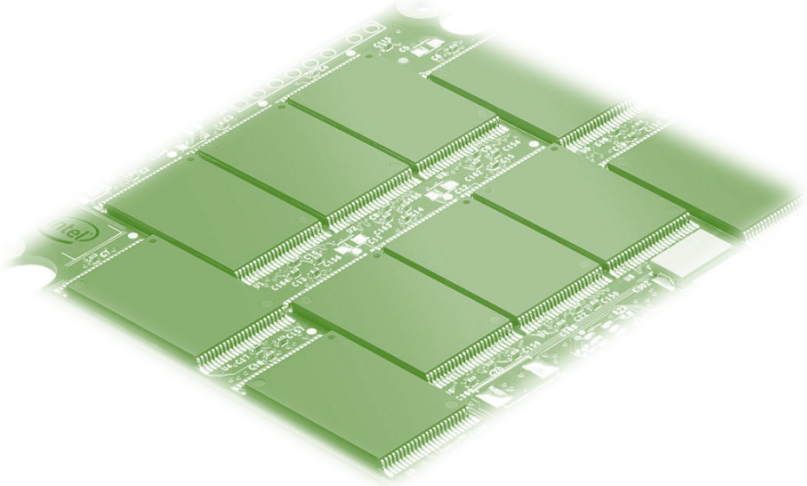
| | VST | Traditional Stress Tests |
|---|---|---|
| **Speed** | Native PC and server speed | Limited by SSD hardware |
| **Scalability** | Multi-sever and multi-core parallelism | Limited by the num. of SSDs |
| **Reproducing a failed stress test** | 100% guarantee | Critical bugs can only appear once! |
| **Investigating a failed stress test** | Software debugging tools | Complicated and expensive equipment |

# Outline

- Background
- Challenges and our contributions
- Virtual Stress Test design
- Evaluation
- Conclusions

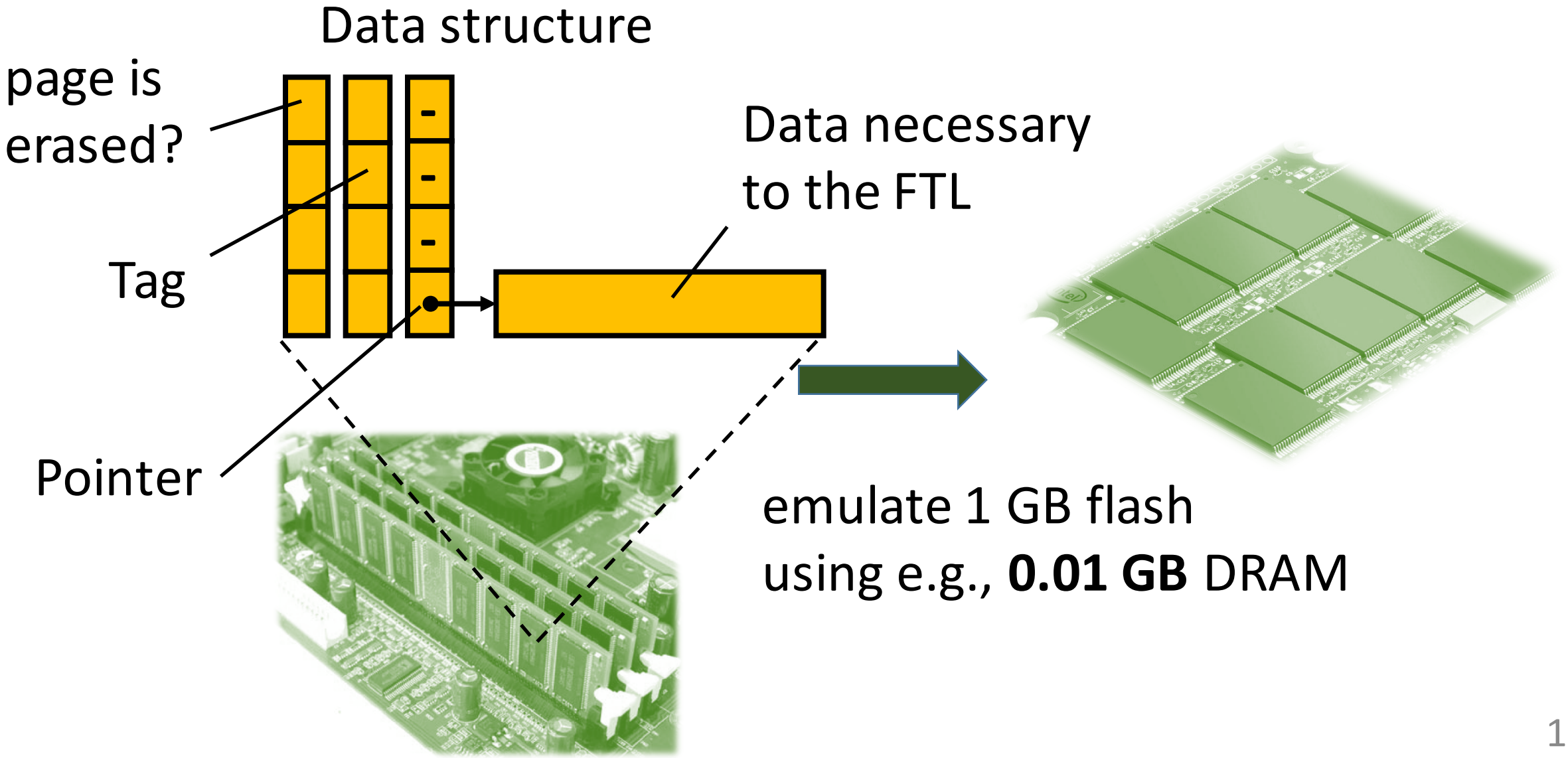
# Naïve Flash Memory Emulation



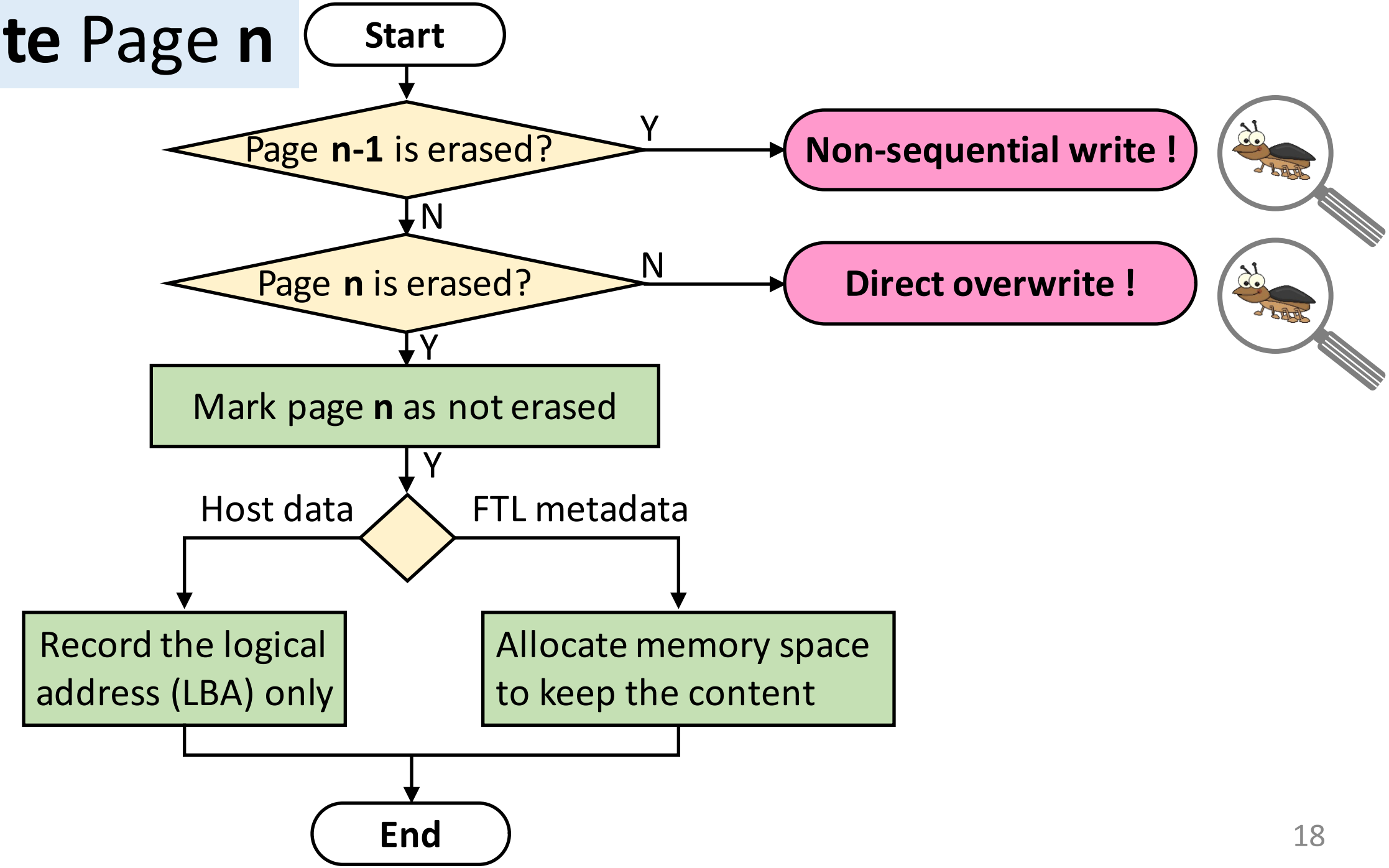



emulate 1 GB flash using 1 GB DRAM

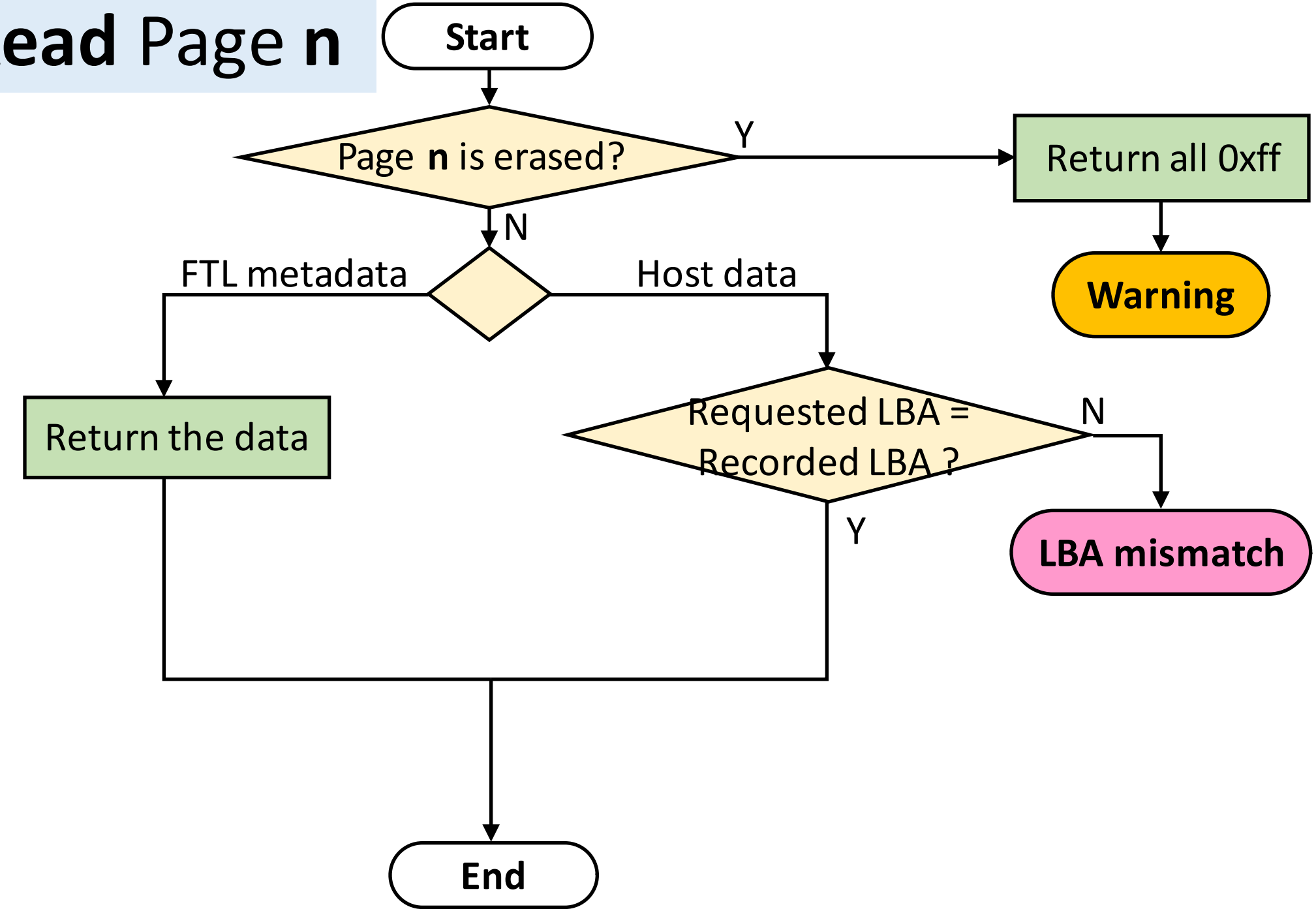
# VST's Flash Memory Emulation



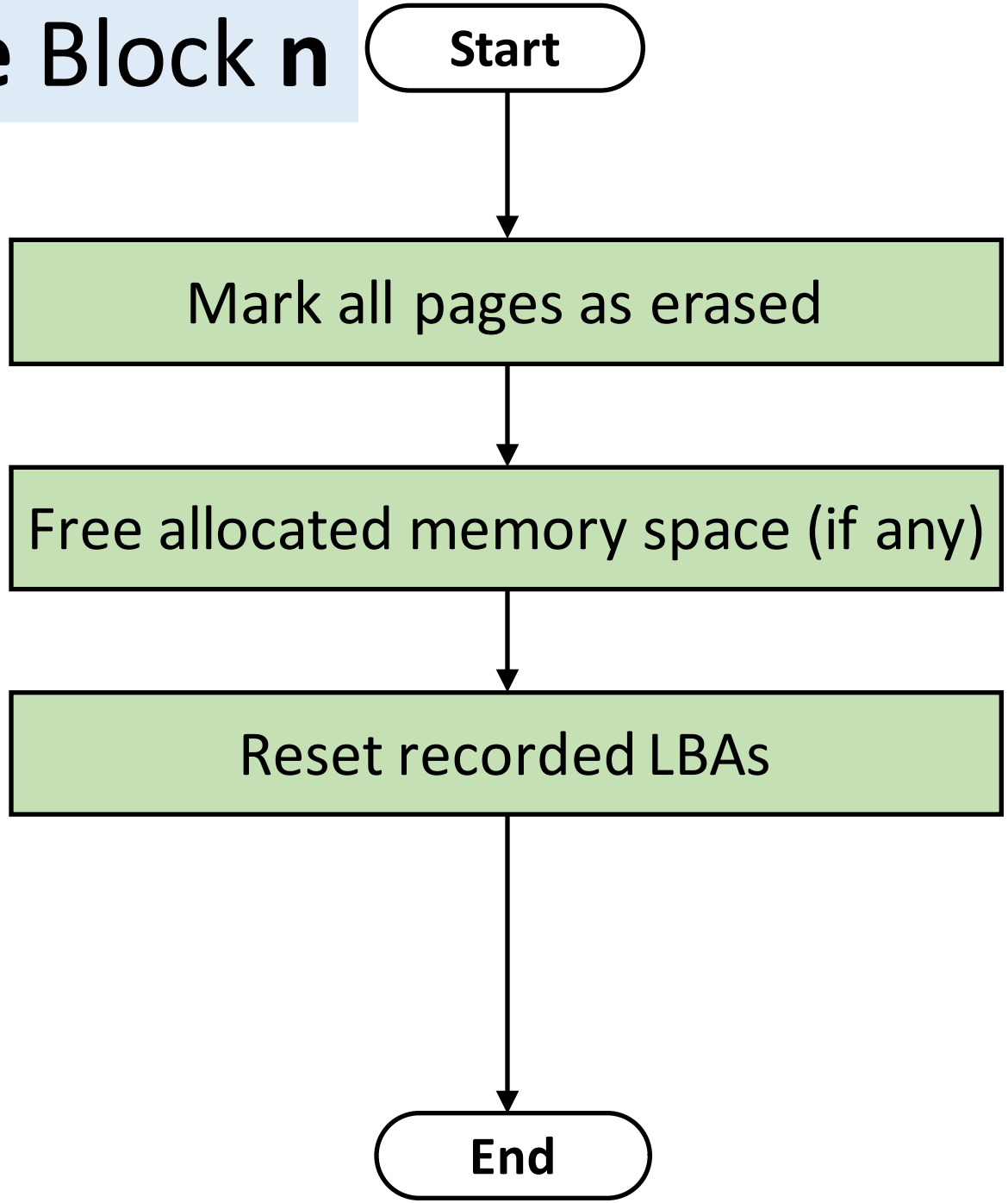
# **Write** Page **n**


Start
Page n-1 is erased?
Y
Non-sequential write !
N
Page n is erased?
N
Direct overwrite !
Y
Mark page n as not erased
Y
Host data
FTL metadata
Record the logical address (LBA) only
Allocate memory space to keep the content
End

# **Read** Page **n**


Start
Page n is erased?
Y
Return all 0xff
Warning
N
FTL metadata
Host data
Return the data
Requested LBA = Recorded LBA ?
N
Y
LBA mismatch
End

# **Erase** Block **n**



# Outline

- Introduction and background
- Challenges and our contributions
- Virtual Stress Test design
- Evaluation
  - Discovered bugs
  - Testing speed
- Conclusions

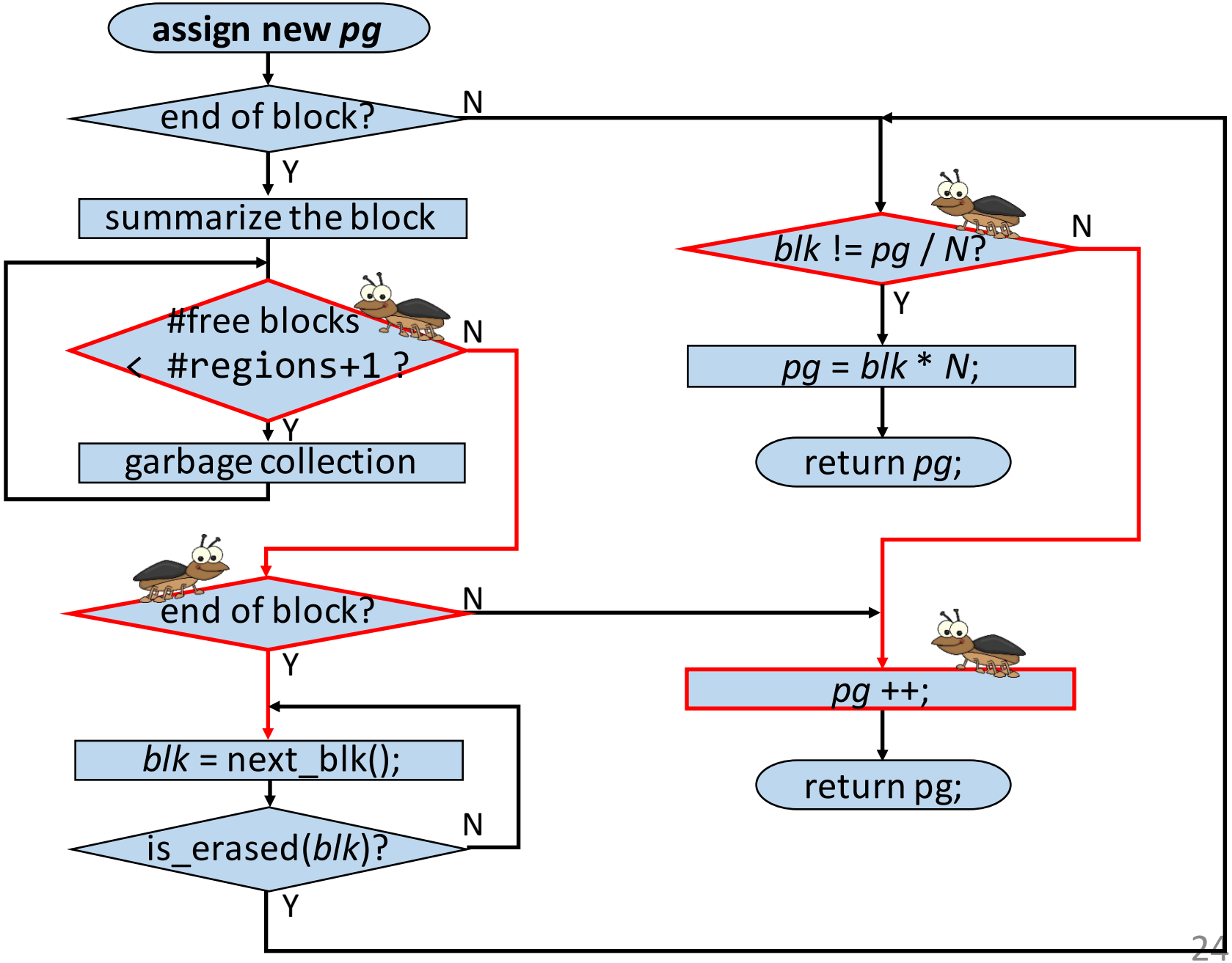
# Experimental Setup

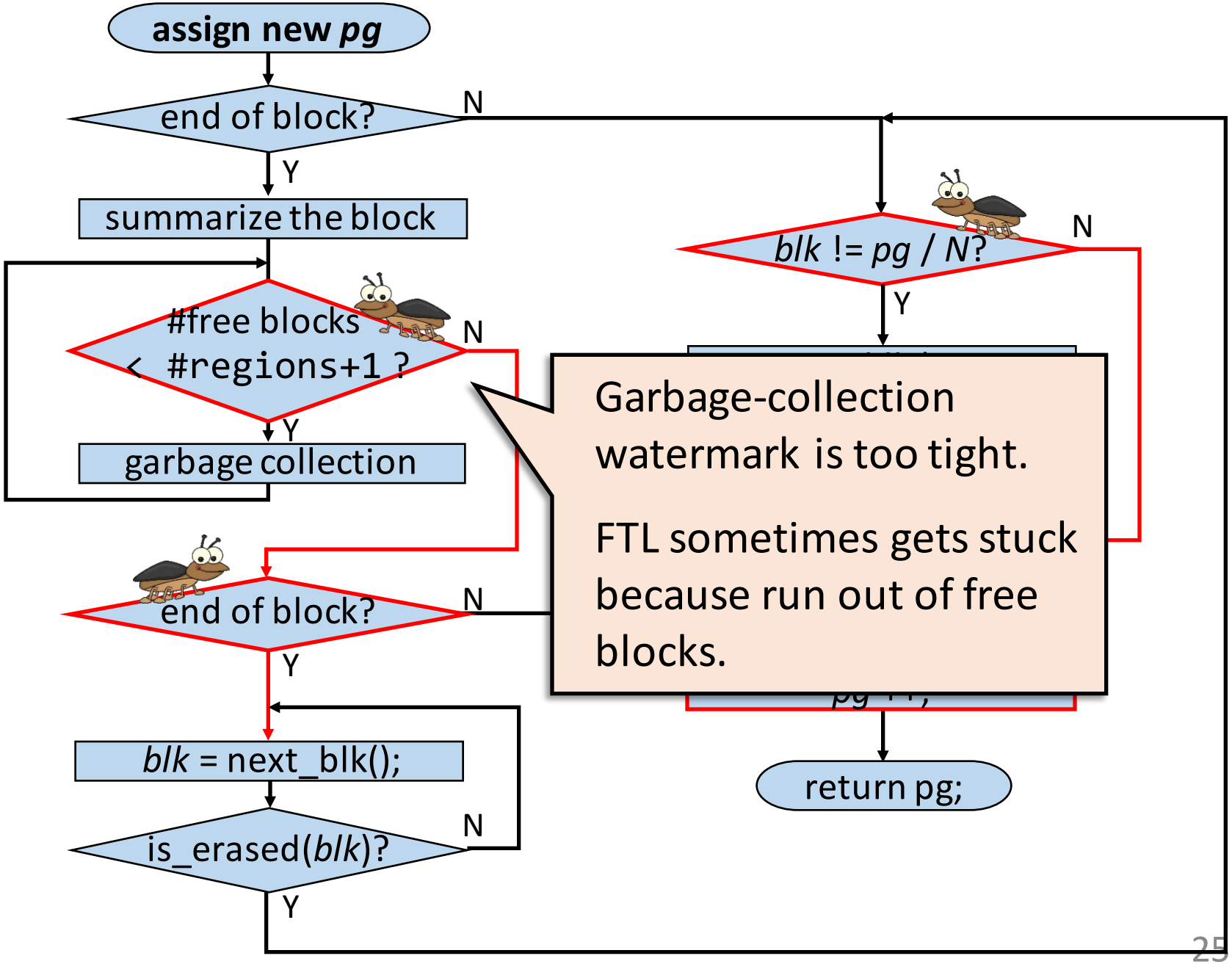
- Three OpenSSD FTLs
  - Greedy, DAC, FASTer
- 16 write-intensive disk traces
  - Each contains 1TB write amount
- Desktop computer
  - Intel i7 3.6GHz 4-core CPU
  - 32GB DRAM

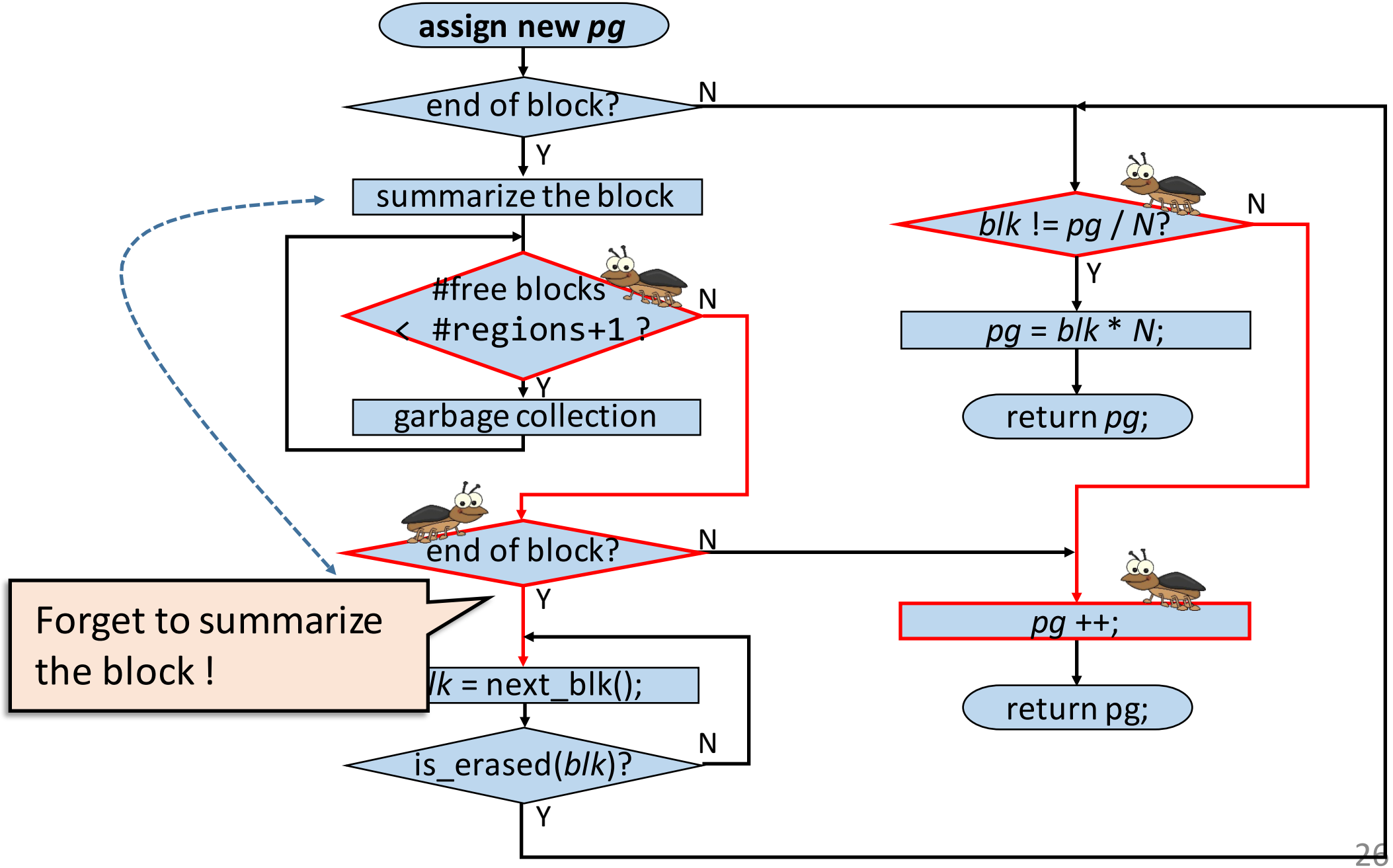
| Name | Write | | Read | |
|---|---|---|---|---|
| | #Requests (M) | Tot. Size (TB) | #Requests (M) | Tot. Size (TB) |
| hm_0 | 78 | 1 | 43 | 0.5 |
| mds_0 | 87 | 1 | 12 | 0.3 |
| prn_0 | 67 | 1 | 8 | 0.2 |
| proj_0 | 23 | 1 | 3 | 0.1 |
| prxy_0 | 91 | 1 | 3 | 0.0 |
| prxy_1 | 60 | 1 | 113 | 1.8 |
| rsrch_0 | 80 | 1 | 8 | 0.1 |
| src1_0 | 18 | 1 | 28 | 1.1 |
| src1_2 | 28 | 1 | 9 | 0.2 |
| src2_0 | 90 | 1 | 11 | 0.1 |
| src2_2 | 18 | 1 | 8 | 0.6 |
| stg_0 | 77 | 1 | 14 | 0.4 |
| ts_0 | 85 | 1 | 18 | 0.3 |
| usr_0 | 69 | 1 | 47 | 2.0 |
| wdev_0 | 84 | 1 | 21 | 0.3 |
| web_0 | 79 | 1 | 34 | 1.1 |

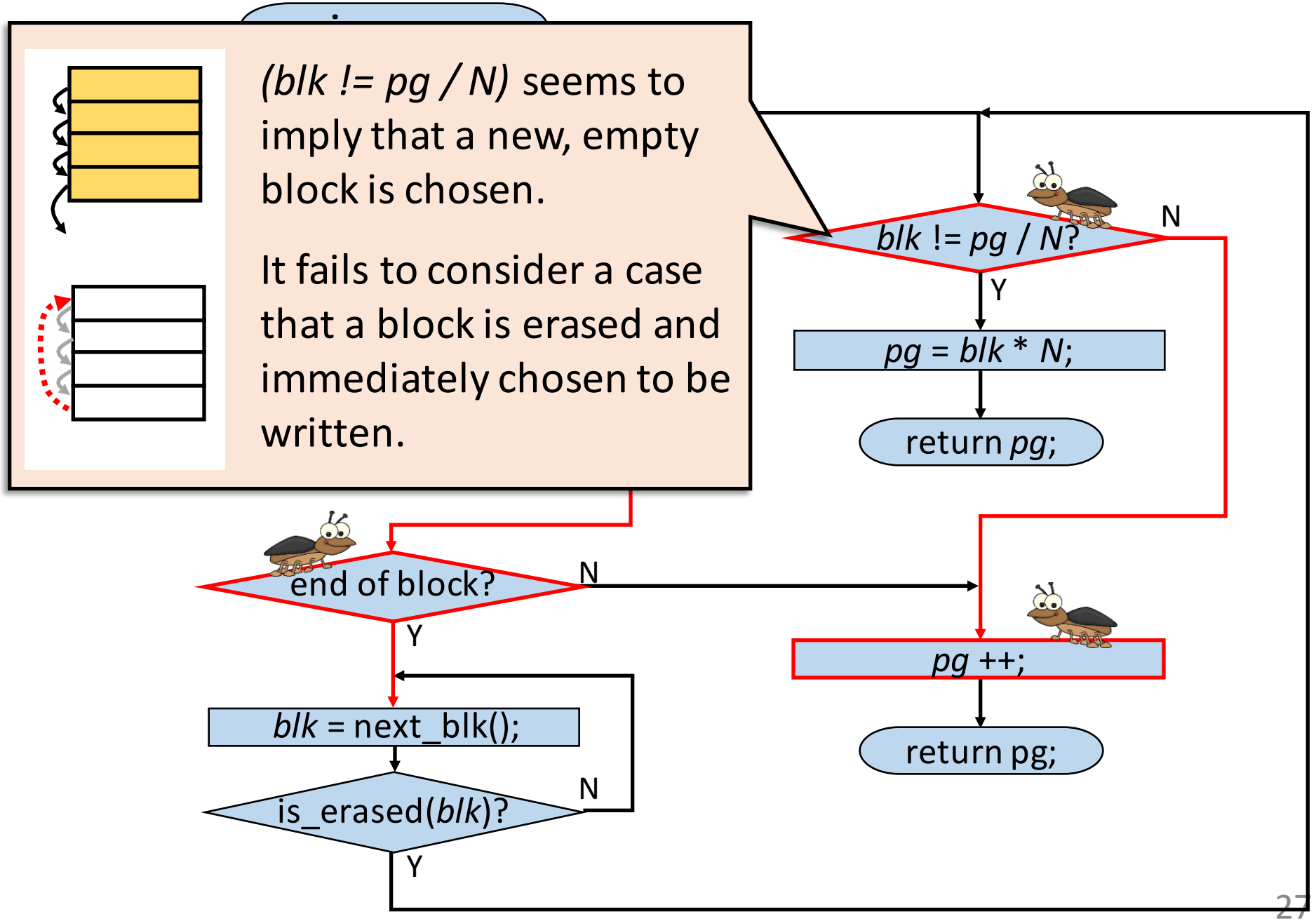
# Bug Detection Results

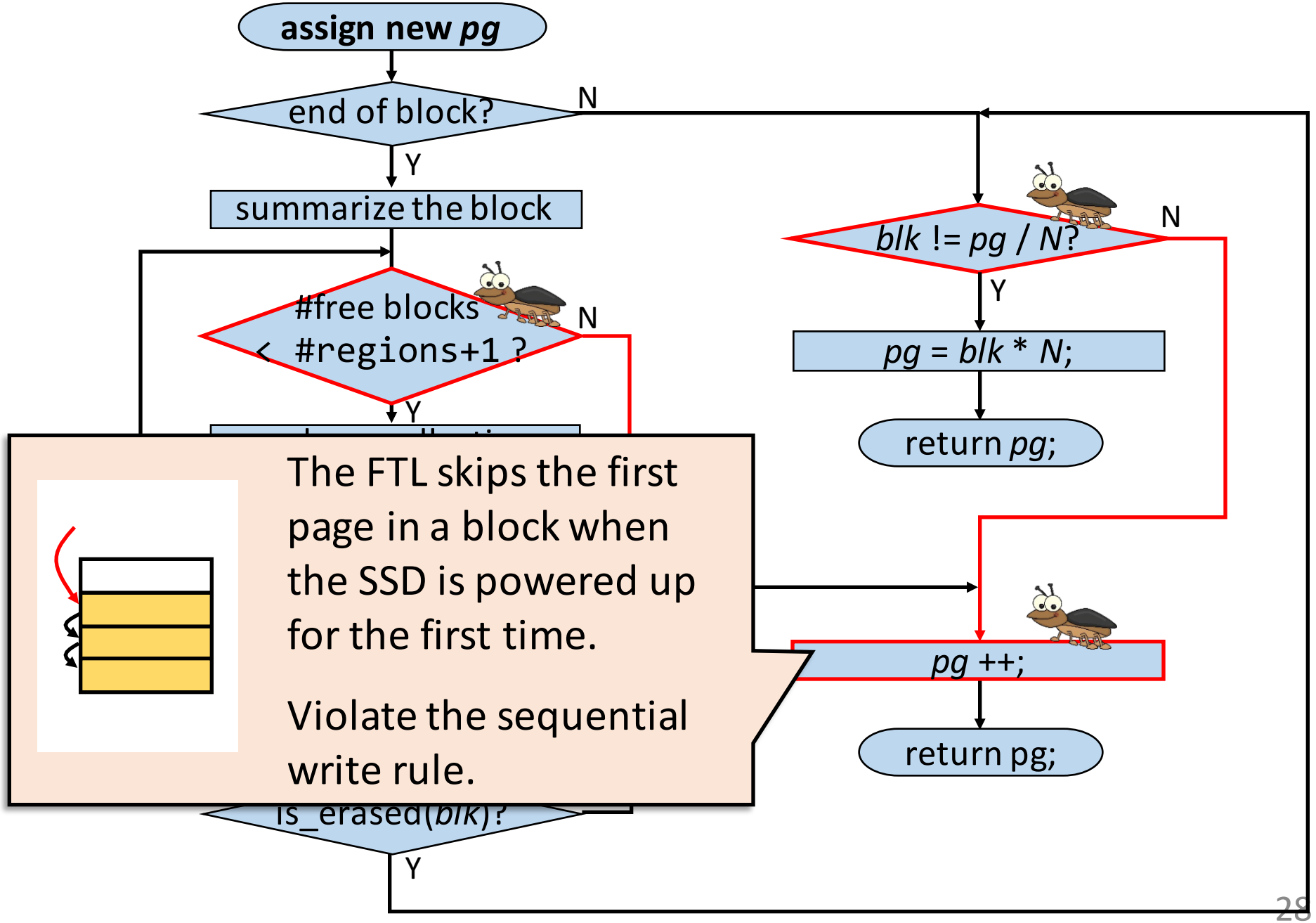
- Seven new bugs are discovered
  - **Four** in DAC FTL
  - **Two** in Greedy FTL
  - **One** in the FASTer FTL
- Briefly describe the bugs in the DAC FTL
- Quantitatively show that the speed and scalability of stress tests are important


assign new pg
end of block?
N
Y
summarize the block
#free blocks
< #regions+1 ?
N
Y
garbage collection
blk != pg / N?
N
Y
pg = blk * N;
return pg;
end of block?
N
Y
pg ++;
return pg;
blk = next_blk();
is_erased(blk)?
N
Y

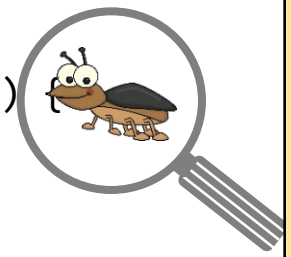

assign new pg
end of block?
N
Y
summarize the block
#free blocks
< #regions+1 ?
N
Y
garbage collection
end of block?
N
Y
blk = next_blk();
is_erased(blk)?
N
Y
blk != pg / N?
N
Y
return pg;
Garbage-collection watermark is too tight.
FTL sometimes gets stuck because run out of free blocks.

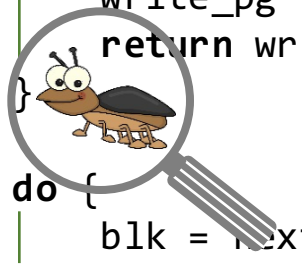

assign new pg
end of block?
N
Y
summarize the block
#free blocks
< #regions+1 ?
N
Y
garbage collection
blk != pg / N?
N
Y
pg = blk * N;
return pg;
end of block?
N
Y
lk = next_blk();
is_erased(blk)?
N
Y
pg ++;
return pg;
Forget to summarize the block !

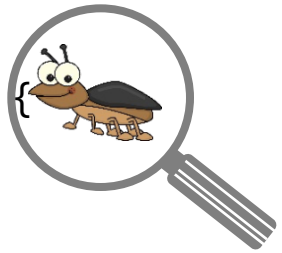

(blk != pg / N) seems to imply that a new, empty block is chosen.
It fails to consider a case that a block is erased and immediately chosen to be written.
blk != pg / N?
N
Y
pg = blk * N;
return pg;
end of block?
N
Y
blk = next_blk();
is_erased(blk)?
N
Y
pg ++;
return pg;

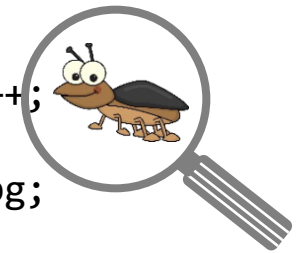

assign new pg
end of block?
N
Y
summarize the block
#free blocks
< #regions+1 ?
N
Y
blk != pg / N?
N
Y
pg = blk * N;
return pg;
The FTL skips the first page in a block when the SSD is powered up for the first time.
Violate the sequential write rule.
pg ++;
return pg;
is_erased(blk)?
Y

```
assign_new_write_vpn(...)
{
    ...
    if (write_pg == the last page of a blk) {
        ...
        summarize the block;
        while (free_blk_cnt() < NUM_REGIONS+1) {
            garbage_collection();
        }

        if (write_pg != the last page of a blk) {
            write_pg ++;
            return write_vpn;
        }

        do {
            blk = next_blk();
        } while (!is_erased(blk));
    }

    if (blk != write_vpn/PAGES_PER_BLK) {
        write_pg = blk * PAGES_PER_BLK;
        ...
    } else {
        write_pg ++;
    }
    return write_pg;
}
```





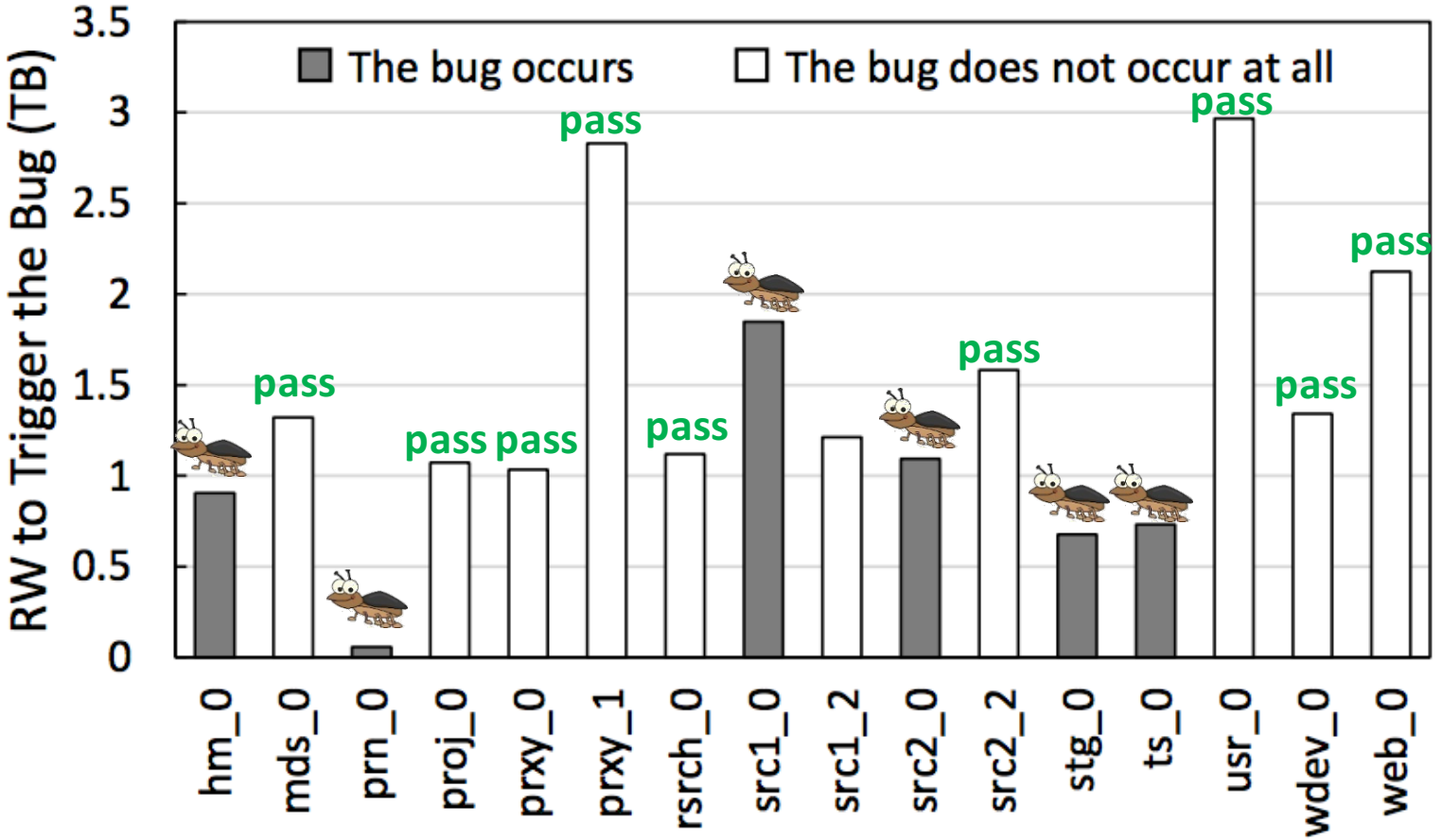




- These four bugs have lurked inside the FTL for many years!
- Speed and scalability advantages of VST are very helpful for us to discover bugs.

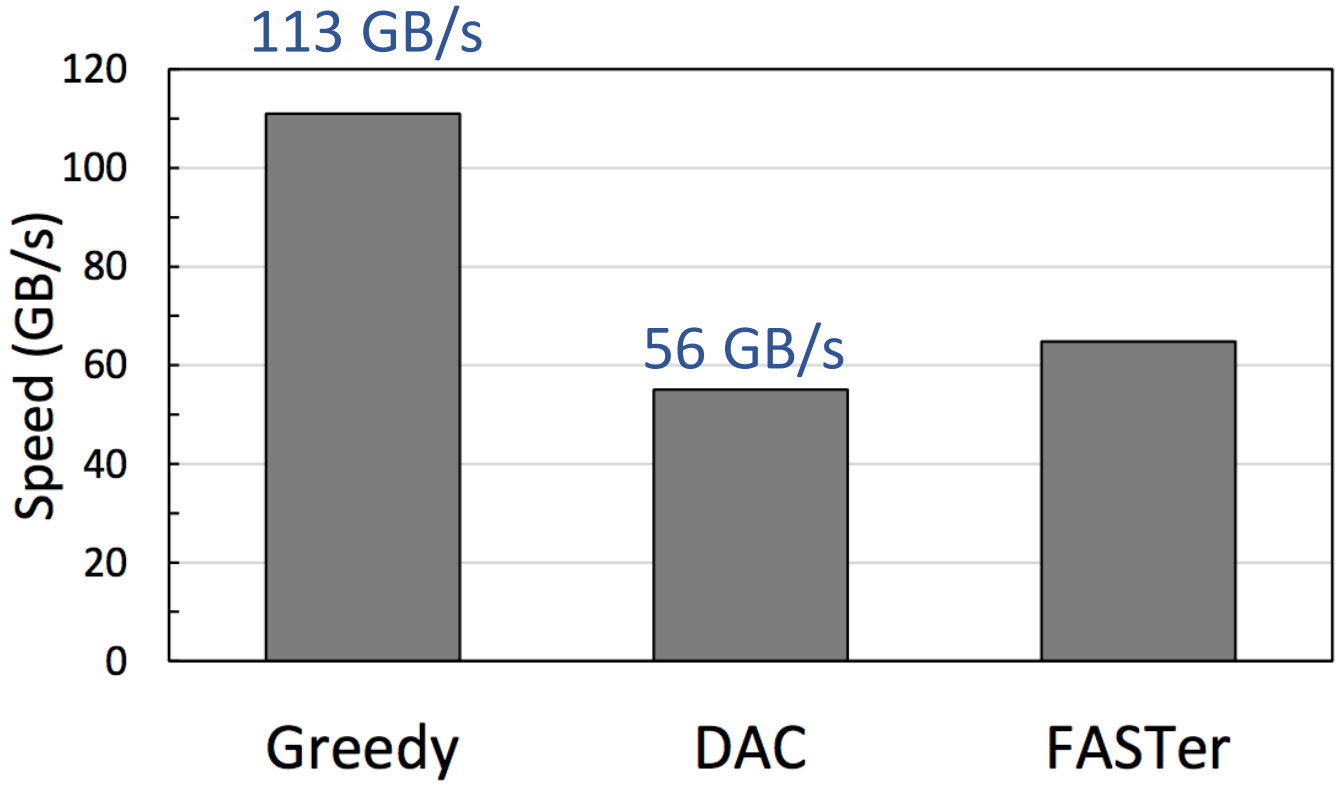
# Importance of Testing Speed and Scalability

- Only 6/16 traces can trigger the bug
- Simply writing 1TB of data into an SSD may overlook this bug
- Fast and scalable stress tests can help engineers to be aware of a bug earlier



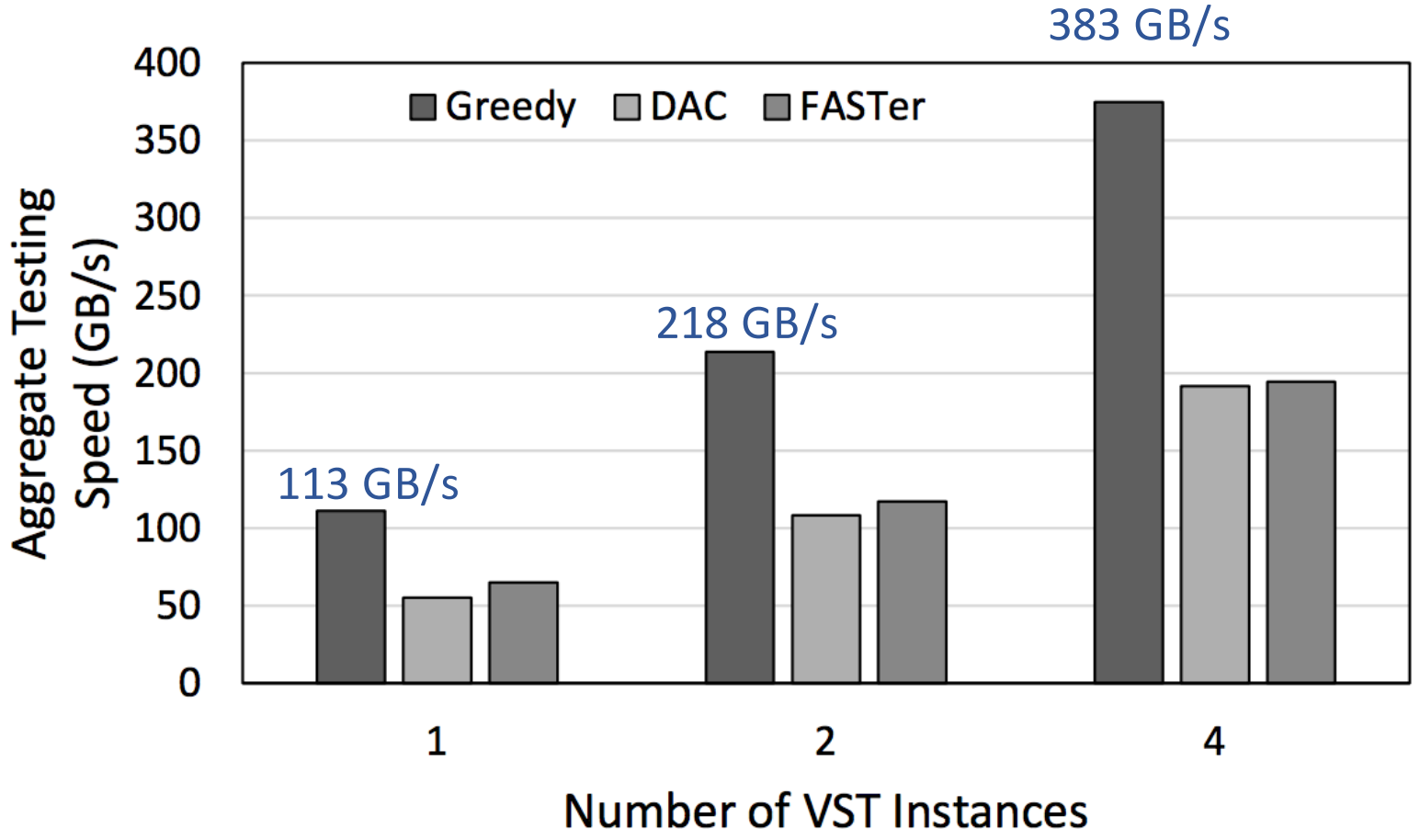
# VST's Strength in High Speed

- Single core
- Three FTLs
- VST achieves 56~113 GB/s on average
- >100X real SSD speed
  - 0.5 ~ 1GB/s



# VST's Strength in High Scalability

- Multiple VSTs can execute in parallel on PCs or servers
- Stress tests is thus easily scalable



# Conclusions

- Virtual Stress Test
  - Stress testing FTLs without real SSDs
- Enhance the SSD/FTL design flow
  - High speed: up to 383 GB/s stress tests, which surpasses SSD and flash speed
  - High scalability: a massive number of stress tests can easily be instantiated
  - Reproducing and investigating failed tests on VST are relatively easy
- We apply VST to OpenSSD and discover seven new bugs

# VST: A Virtual Stress Testing Framework for Discovering Bugs in SSD Flash-Translation Layers