



Open NAND Flash Interface: The First Wave of NAND Standardization

Amber Huffman
Senior Staff Architect

MEMS 002

Intel Developer
FORUM

© 2006 Intel Corporation

Agenda

- **Problem ONFI solves**
- Organization and current status
- Technical details
- What's Ahead

Intel Developer
FORUM



Impact of "Similar" Behavior

- Differences impact time to market and revenue
 - Requires software/firmware updates and thus increases product qualification time

NAND Flash Product Integration

Delayed Opportunities...

Qualification Time

NAND	Vendor Samples New NAND	Software / Firmware Updates	NAND Driver Testing	Product Level Testing	Ship New NAND
HDD	Supplier Samples New HDD	Product Level Testing	Ship New HDD	← Goal for NAND qualification time	

- NAND Flash requires more qualification time than other commodity memory products
- Lost revenue opportunity that could be rectified with standard interface

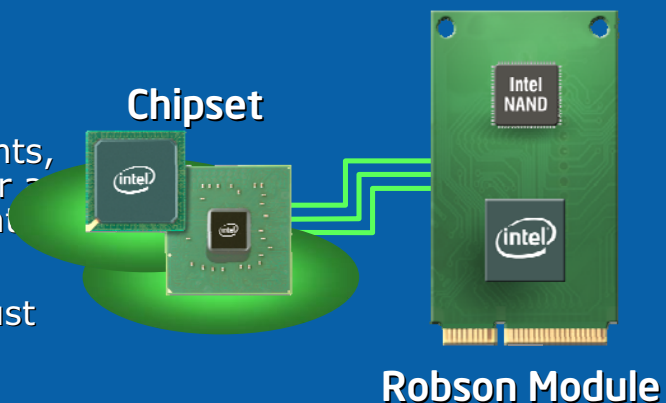
Differing implementations of NAND Flash interface impact time to market and revenue

Intel Developer FORUM Micron intel

5

NAND in Compute Platforms

- NAND is an increasingly important platform ingredient (like DRAM)
 - Intel's platforms have longevity and address numerous market segments, which requires support for a range of NAND components
 - Intel's stable platform must have the means to easily support the latest Flash components



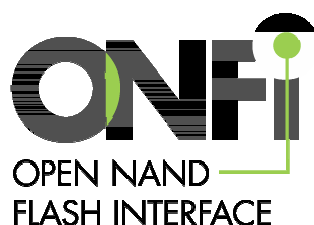
Lack of standard impacts host ability to easily support a range of NAND devices

Agenda

- Problem ONFI solves
- **Organization and current status**
- Technical details
- What's Ahead

Open NAND Flash Interface (ONFI)

- ONFI establishes a standard interface for NAND
- New paradigm: Host vendors can develop for features that they will support and enable them "on the fly" if the NAND supports it
 - Old paradigm: Host vendors wait for the latest NAND part and then start making changes to communicate with the base NAND
- ONFI paves the way to:
 - Shorter NAND and product qualification times
 - More innovation since NAND command sequences always work



SONY

hynix



PHISON
Knows What You Need

High Level Features in ONFI 1.0

- Discoverable capabilities
 - Provides parameter page that describes NAND capabilities
- Multiple LUN operations
 - Enables independent operations
- Interleaved operations
 - Enables operations of the same type to execute simultaneously on the same LUN
- Caching operations
 - Enables data transfer while array operations are ongoing
- Copyback for internal data moves
- Standard timing modes

Command	O/M	1 st Cycle	2 nd Cycle
Read	M	00h	30h
Copyback Read	O	00h	35h
Change Read Column	M	05h	E0h
Read Cache Enhanced	O	00h	31h
Read Cache	O	31h	
Read Cache End	O	3Fh	
Block Erase	M	60h	D0h
Interleaved	O		D1h
Read Status	M	70h	
Read Status Enhanced	O	78h	
Page Program	M	80h	10h
Interleaved	O		11h
Page Cache Program	O	80h	15h
Copyback Program	O	85h	10h
Change Write Column	M	85h	
Read ID	M	90h	
Read Parameter Page	M	ECh	
Read Unique ID	O	EDh	
Get Features	O	EEh	
Set Features	O	EFh	
Reset	M	FFh	

ONFI 1.0 Status Update

- Workgroup has delivered 0.9 draft of the ONFI specification
- Final item to polish is the state machine describing ONFI behavioral requirements
 - Describes externally visible behavioral requirements between the host and the NAND

L_Idle_TargetRequest	If Target indicates an address, the address is stored by the LUN.
1. Target indicates WP# value	→ L_WP_Update
2. Target requests SR register update	→ L_SR_Update
3. Target requests status or status command received	→ L_Status_Execute
4. Target requests page register clear	→ L_Idle_ClearPageReg
5. Target requests page register invalidate	→ L_Idle_InvalidPageReg
6. Target indicates Program request for this LUN	→ L_PP_Execute
7. Target indicates Erase request for this LUN	→ L_BE_WaitForCmd
8. Target indicates Read request for this LUN	→ L_RD_WaitForCmd
9. Target indicates Read Unique ID request	→ L_Idle_RdUid

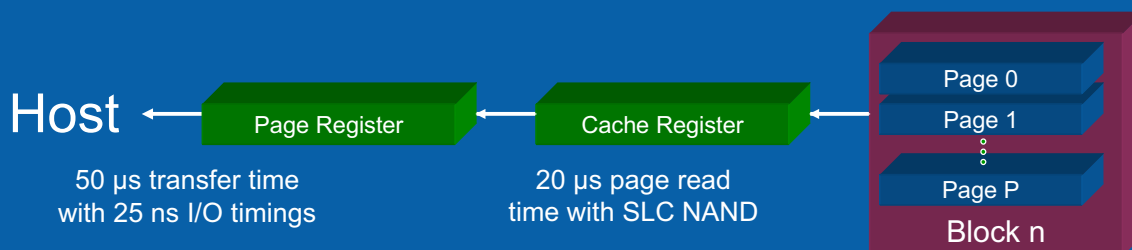
ONFI 1.0 specification to be published in January with full set of features

Agenda

- Problem ONFI solves
- Organization and current status
- **Technical details**
- What's Ahead

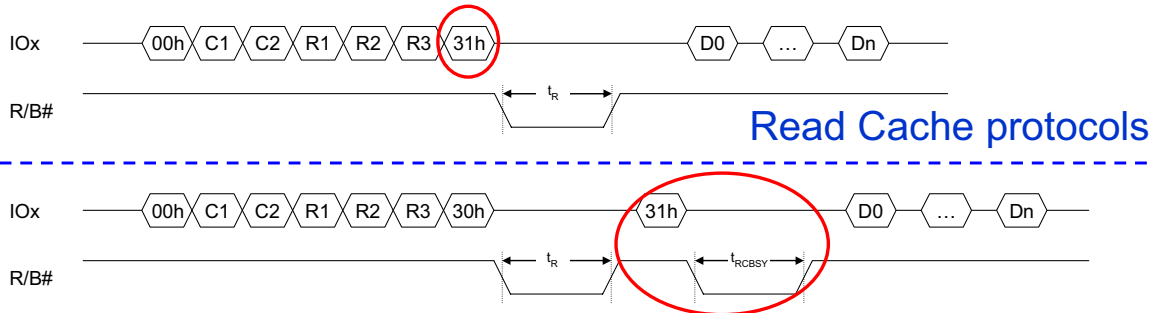
Enhancing Read Performance

- The host cannot directly read from the NAND array
- Reads happen in two steps:
 - 1) Data flows from the array to the page register
 - 2) Data flows from the page register to the host
- NAND IHVs introduced a cache register to eliminate lock-step reliance on page register for multiple read commands
- Allows NAND to read the next page from the array while transferring the current page to the host



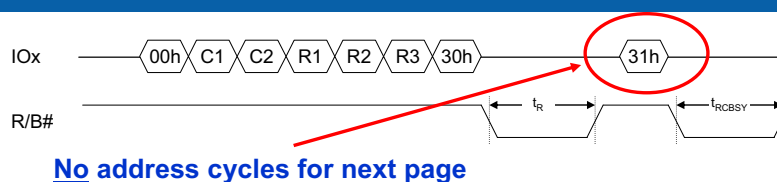
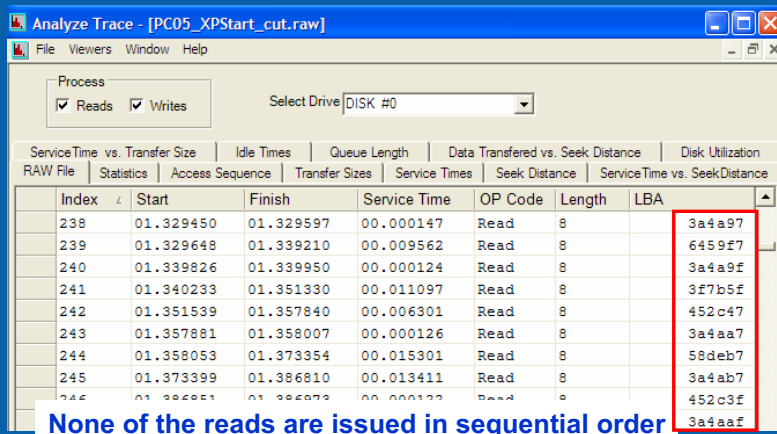
Read Cache command divergence

- In industry today, there are two Read Cache command variants in wide use
- Read Cache protocol 1:
 - Issue a modified Read command (finish with 31h instead of 30h)
 - Each time the end of a page is reached, the NAND automatically starts reading the next page
- Read Cache protocol 2:
 - Issue a normal read command and wait the t_R array access time
 - Issue a 31h command to tell the NAND to start a background read operation



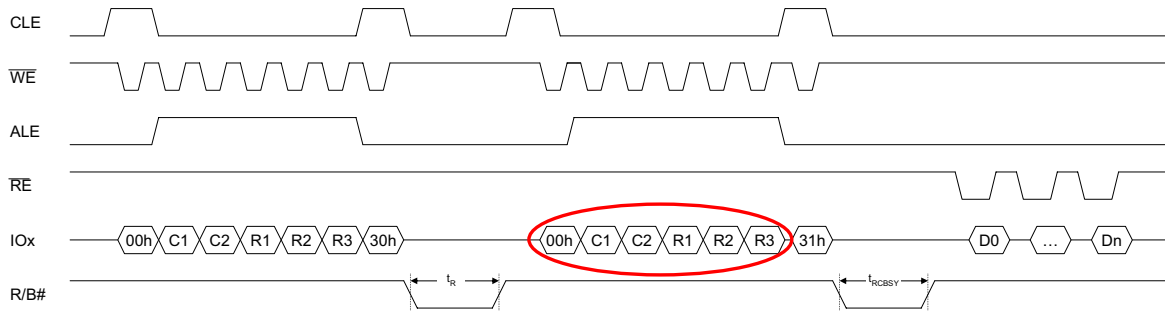
Read Cache applicability to high performance applications

- Applications, like Robson, receive frequent small random reads
- Read Cache seems ideal for read sequence shown
- *Gotcha:* Read Cache is only for sequential access
- No address cycles provided for subsequent pages in either protocol



ONFI delivers Read Cache Enhanced

- ONFI enhances Read Cache to allow the host to specify the address for each cached read
- If no address provided, defaults to sequential access to preserve any previous host investment in Read Cache



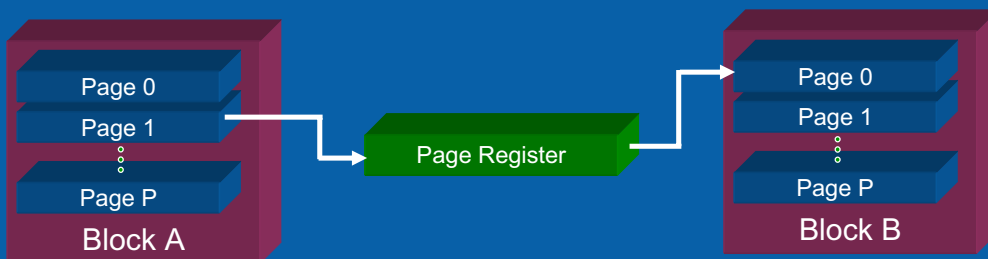
Performance Benefit of Read Cache Enhanced

- Read Cache Enhanced delivers dramatic performance benefits in any read sequence (sequential or random)
 - For 25 ns I/O timings, benefit is **29.6%**
 - For 20 ns I/O timings, benefit is **36.5%**
- Benefit **grows** with faster I/O timings

	Read Performance, 25 ns I/O timings	Read Performance, 20 ns I/O timings
Normal page read	28.94 MB/s	33.86 MB/s
Read Cache, sequential	37.62 MB/s	46.34 MB/s
Read Cache, random	37.52 MB/s	46.22 MB/s

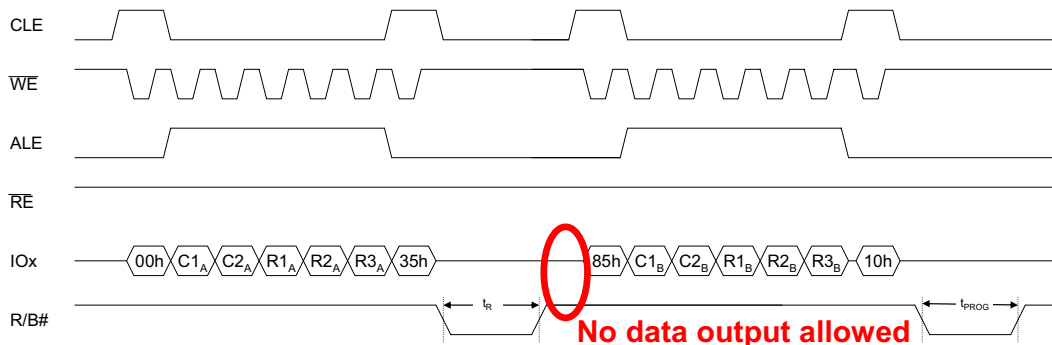
Copyback for Data Moves

- Copyback is a common NAND command used to move data from one block to another
 - Example: Move page 1 in A to page 0 in B
- Used for relocations and other operations where the host does not need to read the data
 - Note: The host can modify the data in the page register



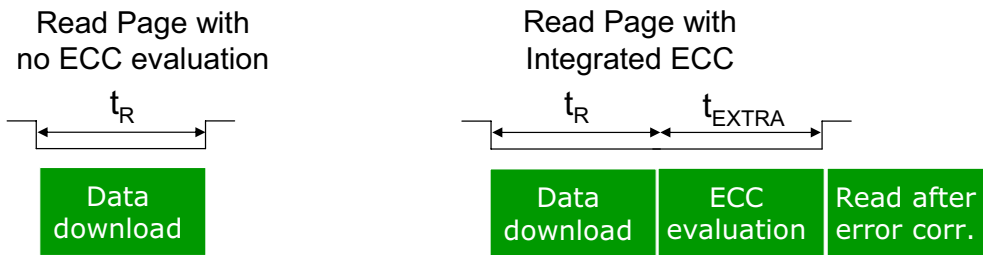
Data Integrity and Copyback

- Copyback does not allow the host to read the data
- If there is an ECC error on the read from the original location, the error is **propagated**
- Use of traditional copyback reduces data integrity



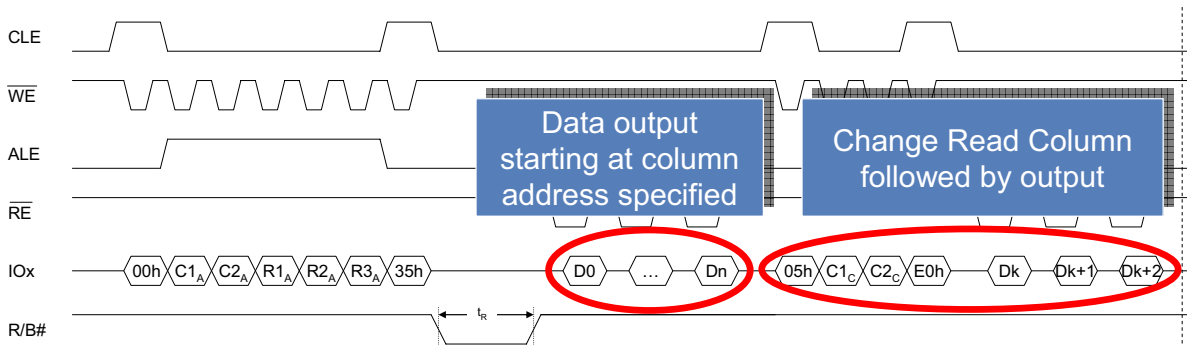
Using Integrated ECC to Resolve Data Integrity concern

- One approach to resolving the data integrity concern is to integrate ECC in the NAND for copyback operations
 - Increases read busy time by t_{EXTRA} to allow ECC evaluation
- Issues:
 - ECC NAND uses may not suit host application (e.g. too weak)
 - Time to evaluate in NAND is very likely longer than the host to do the same job
 - Integrated ECC increases the NAND die area



Copyback in ONFI

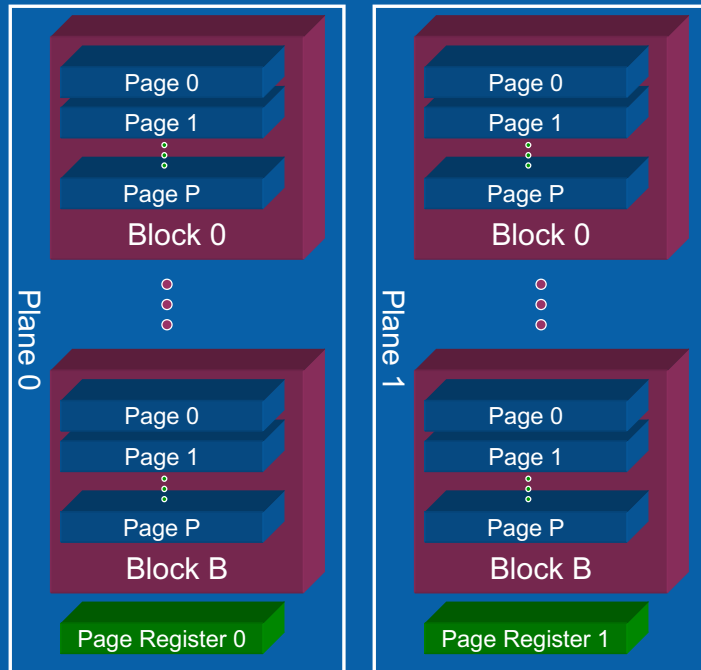
- ONFI resolves concern by making data output a mandatory part of Copyback
- Hosts that desire high data integrity can use ECC of their choice



Traditional Copyback after "A" ^A

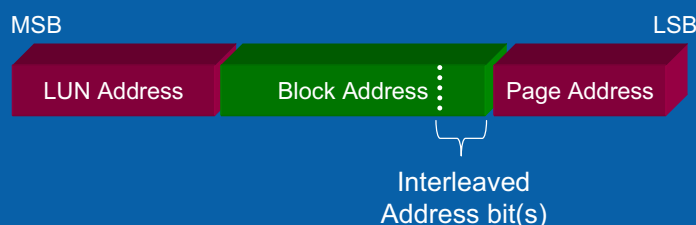
Increasing Concurrency for NAND Array Operations

- NAND vendors have started splitting the array into “planes” within a die
- Allows simultaneous operations of the same type to different blocks
 - Reads
 - Programs
 - Erases



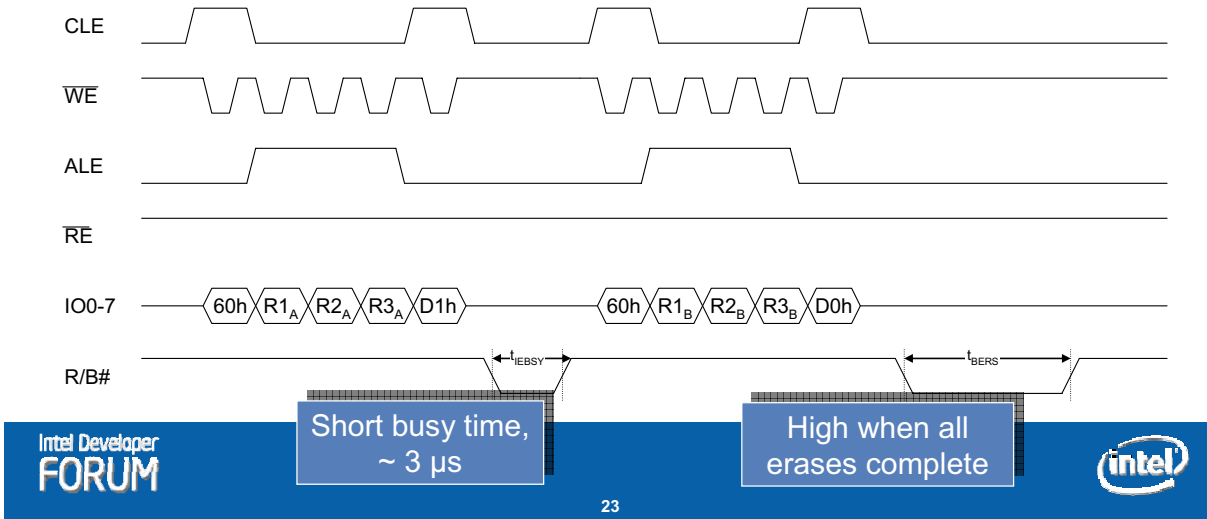
Interleaved Operations

- To avoid dependence on NAND implementation and allow innovation, ONFI abstracts out multi-plane operations
- Interleaved operations are dependent operations allowed for different blocks on a logical unit
 - Interleaved address is LSB bit(s) of the block address
- Interleaving “types”:
 - Concurrent interleaving: Array operations for all blocks start after final command and execute in parallel
 - Overlapped interleaving: Array operations are independent and start after the operation is issued



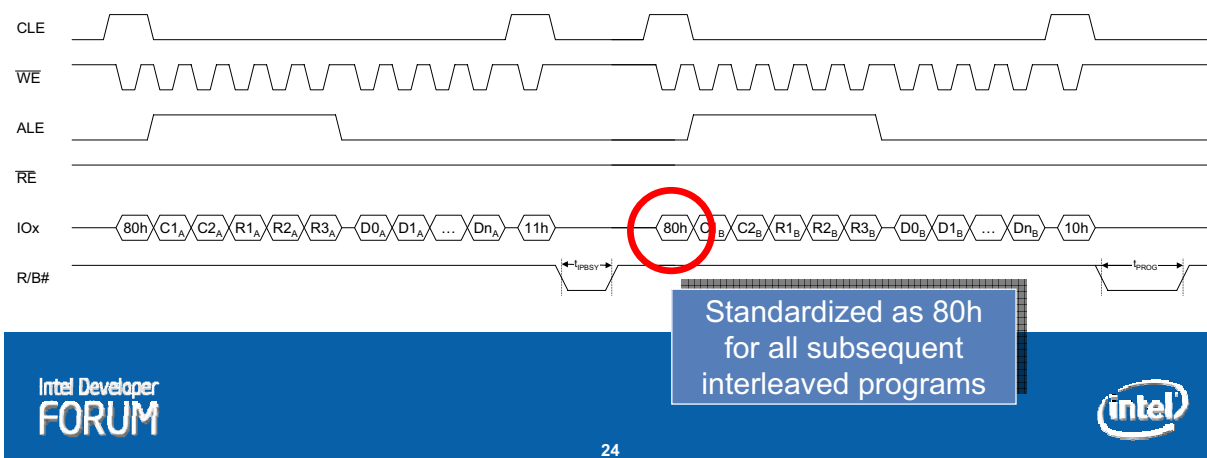
Interleaved Erase Operations

- Sequence for concurrent and overlapped interleaved erases identical
- Easily extensible to more interleaved addresses with additional 60h/D1h sequences



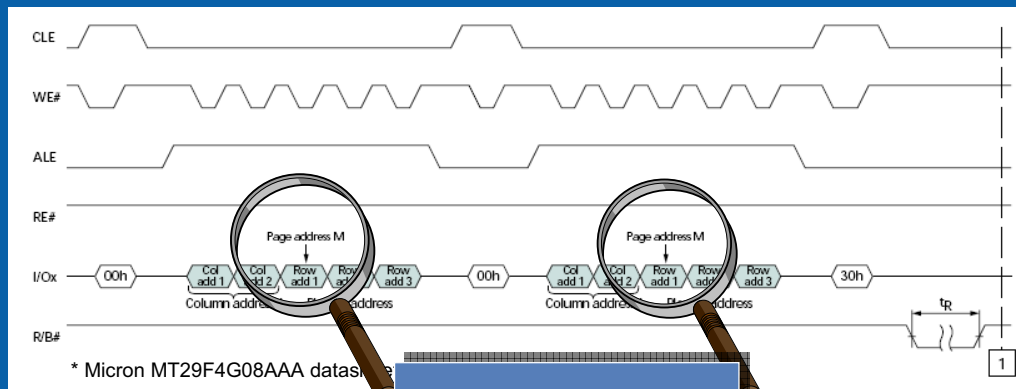
Interleaved Program Operations

- ONFI standardizes subsequent program commands as 80h
 - Some multi-plane program operations have 81h
 - No value in 81h, 82h, 83h, etc since previous ending command cycle of 10h or 11h provides necessary context
- Allows host to generically support n-interleaved addresses



Interleaved and Multi-Plane Limitations impact on Reads

- Interleaved operations and multi-plane operations have a significant limitation
 - The page address is required to be the same



* Micron MT29F4G08AAA datasheet

“Page address M”

Intel Developer FORUM

*Other names and brands may be claimed as the property of others

25

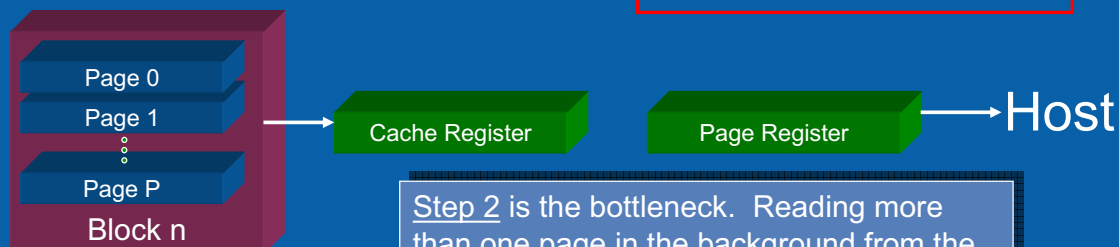


ONFI does not support Interleaved Reads

- ONFI has chosen to not implement interleaved reads
- Why?
 - Limitation of same page address makes utility low
 - Rife with bus contention issues
 - Read Cache Enhanced provides all of the benefits

Step 1: Array to Cache Register
20 microseconds (SLC)

Step 2: Page Register to Host
50 microseconds (25 ns timings)



Step 2 is the bottleneck. Reading more than one page in the background from the array is pointless. Still serialized to host...

Intel Developer FORUM

26



ONFI Technical Details Summary

- ONFI enhances Read Cache to provide value in all read sequences
- ONFI removes data integrity concerns from Copyback with ECC flexibility
- ONFI provides high performance interleaved operation sequences
 - Allows for concurrent or overlapped array operations with a single host sequence
 - Plans for n-interleaved operations for the future

**ONFI simplifies command sequences
while enhancing value**

Agenda

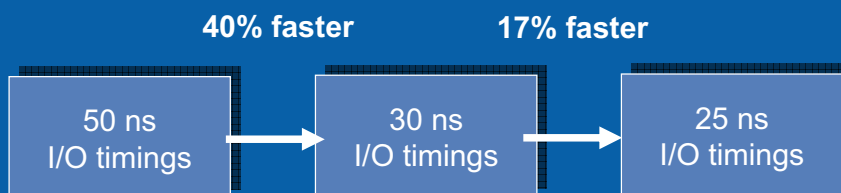
- Problem ONFI solves
- Organization and current status
- Technical details
- **What's Ahead**

Where is ONFI Going?

- Standardization for existing NAND protocol was the first phase of ONFI
- NAND has enormous opportunities for innovation, which the ONFI Workgroup is evaluating
- The following foils show a flavor of the potential topics ONFI may pursue moving forward
 - I/O buffer timing enhancements
 - NAND connection in the platform
 - Block Abstracted NAND

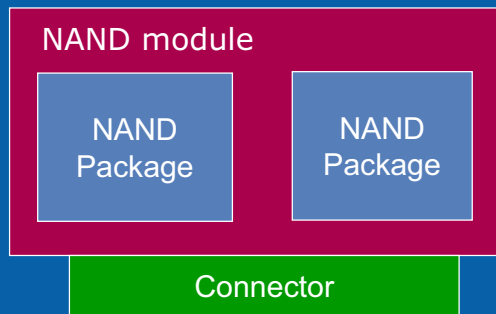
I/O Buffer Timing Enhancements

- NAND timing is not source synchronous, making it difficult for the host to latch the data cleanly at higher speeds
- This difficulty is reflected in the slowdown in NAND timing improvements
- Enhancements beyond “going faster” will need to be made to get good scaling



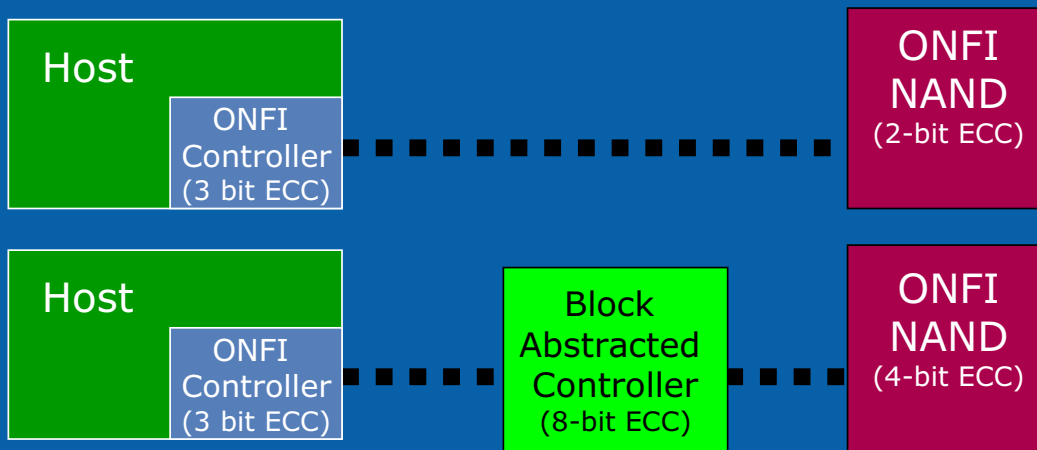
NAND Connection in the Platform

- NAND is penetrating the PC platform
- There is not a standard connector to enable attachment of NAND in the PC
 - I.e., there is nothing like a DIMM for NAND



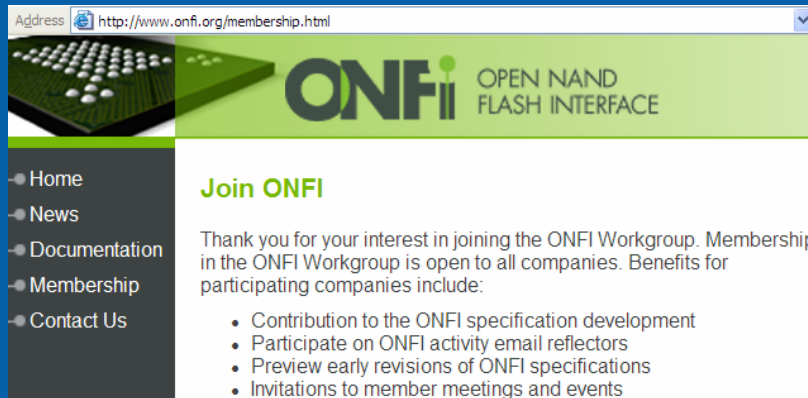
Block Abstracted NAND to enable Broader NAND Use

- NAND may have ECC or other management requirements that are beyond the host's capabilities
- Block Abstracted NAND allows a controller to be inserted in the middle that abstracts some of the complexities of NAND



Join ONFI to Help Drive Future Innovation

- The ONFI Workgroup will be firming up the ONFI feature roadmap in the next month



Join ONFI to deliver more tangible benefits in key areas to the NAND industry

Intel Developer
FORUM



33

Summary

- Lack of standard impacts host ability to easily support a range of NAND devices
- ONFI 1.0 specification to be published in January with full set of features
- ONFI simplifies command sequences while enhancing value
- Join ONFI to deliver more tangible benefits in key areas to the NAND industry

Join ONFI to catch the next wave of NAND standardization

Intel Developer
FORUM



34

Additional sources of information on this topic:

- MEMC001 Chalk talk
- More web based info: www.onfi.org

Please fill out the Session Evaluation Form

Session presentation available in Content Catalog on the IDF web site –

when prompted enter:

Username: idf

Password: fall2006

(Please note these are case sensitive)

Thank You for your input, we use it to improve future Intel Developer Forums

Join us at the Spring 2007 IDF on March 20-22 in San Francisco!!



Intel Developer
FORUM

