A Standard Interface for NAND Flash

Amber Huffman
Senior Staff Architect, Intel

Michael Abraham
Sr. Applications Engineer, Micron
Legal Disclaimer

INFORMATION IN THIS DOCUMENT IS PROVIDED IN CONNECTION WITH INTEL® PRODUCTS. NO LICENSE, EXPRESS OR IMPLIED, BY ESTOPPEL OR OTHERWISE, TO ANY INTELLECTUAL PROPERTY RIGHTS IS GRANTED BY THIS DOCUMENT. EXCEPT AS PROVIDED IN INTEL’S TERMS AND CONDITIONS OF SALE FOR SUCH PRODUCTS, INTEL ASSUMES NO LIABILITY WHATSOEVER, AND INTEL DISCLAIMS ANY EXPRESS OR IMPLIED WARRANTY, RELATING TO SALE AND/OR USE OF INTEL PRODUCTS INCLUDING LIABILITY OR WARRANTIES RELATING TO FITNESS FOR A PARTICULAR PURPOSE, MERCHANTABILITY, OR INFRINGEMENT OF ANY PATENT, COPYRIGHT OR OTHER INTELLECTUAL PROPERTY RIGHT. Intel products are not intended for use in medical, life saving, or life sustaining applications.

Intel may make changes to specifications and product descriptions at any time, without notice. Designers must not rely on the absence or characteristics of any features or instructions marked "reserved" or "undefined." Intel reserves these for future definition and shall have no responsibility whatsoever for conflicts or incompatibilities arising from future changes to them.

Contact your local Intel sales office or your distributor to obtain the latest specifications and before placing your product order.

This document contains information on products in the design phase of development. Do not finalize a design with this information. Revised information will be published when the product is available. Verify with your local sales office that you have the latest datasheet before finalizing a design.

Robson and other code names featured are used internally within Intel to identify products that are in development and not yet publicly announced for release. Customers, licensees and other third parties are not authorized by Intel to use code names in advertising, promotion or marketing of any product or services and any such use of Intel’s internal code names is at the sole risk of the user.

All products, dates, and figures specified are preliminary based on current expectations, provided for planning purposes only and are subject to change. All products, dates, and figures specified are preliminary based on current expectations, provided for planning purposes only, and are subject to change without notice.

Intel and the Intel logo is a trademark or registered trademark of Intel Corporation or its subsidiaries in the United States and other countries.

*Other names and brands are the property of their respective owners.

Copyright © 2006, Intel Corporation
Agenda

NAND Flash Product Integration

System problems with NAND Today

Open NAND Flash Interface (ONFI) initiative overview

ONFI Technical Preview
NAND Flash Is Becoming a Commodity Memory Product

Total NAND Market Segment (Billions of US$)

- 2005: $10.2B
- 2007: $15.2B
- 2009: $20.9B

Source: WSTS, Micron Market Research
NAND Flash Product Integration

Product Update Cycle for HDD

MP3 Player

HDD
30 GB

MP3 Player

HDD
60 GB

No software changes required
Product-level testing and validation focus
Physical and logical interface standard
Product Update Cycle for NAND

Firmware/software changes required
- Change device ID table to support new component
- Add support for new commands to maintain or increase performance

Complete re-test of firmware and/or software
Customer Challenges

Current NAND Flash device identification provides few, if any, details regarding the device’s capabilities.

What is needed to update a product to support new NAND Flash devices (from same or other vendor)?

- **Architecture**
  - Addressing into the NAND Flash memory array
  - ECC requirements
- **Command set**
  - Basic commands
  - Cache program
  - Cache read
  - Parallel operations
  - Status commands and status register bit definitions
- **I/O timing**

Why are there differences?
A Push for Higher Performance

Support parallel operations
- Array architecture
- Addressing
- Command set and status methods

Faster I/O timing

Faster array operations

Increase page size
A Push for Higher Density

Multi-Level Cell (MLC) technology
- Store 2 bits per memory cell instead of 1
- Result
  - Addressing change
  - Decreased performance
  - Reduced cell endurance
  - Stronger ECC requirements – 4 or more bits per 512 bytes
  - Lower cost per bit
Long Product Update Cycles

Vendors developing features to support higher performance and density at the same time in isolation naturally leads to *product inconsistency* causing headaches for customers...

- Commands for new features are inconsistent
- I/O timing (clock cycle, setup, hold times) is inconsistent
- Addressing schemes are inconsistent

Unlike other commodity storage memories, NAND Flash does not have an industry-wide specification that provides product consistency in basic device behaviors.
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.
Agenda

NAND Flash Product Integration
System problems with NAND Today
Open NAND Flash Interface (ONFI) initiative overview
ONFI Technical Preview
Intel’s NAND Dilemma...

NAND Flash is an increasingly important Intel platform ingredient

- Robson technology highlights increasing role of NAND Flash in Intel’s platform plans
- Intel’s platforms have longevity and address numerous market segments, which requires support for a range of NAND Flash components
- Intel’s stable platform must have the means for conveniently supporting the latest Flash components without having to be revamped

Current similar NAND Flash components do not allow for range of NAND to be accommodated in a platform
Similar: Basic Commands

Basic commands *typically* common

- Reset, Read ID, Read, Page Program, Erase, ...

More complex operations all over the map

**Table 1. Command Sets**

<table>
<thead>
<tr>
<th>Function</th>
<th>1st. Cycle</th>
<th>2nd. Cycle</th>
</tr>
</thead>
<tbody>
<tr>
<td>Read</td>
<td>00h</td>
<td>30h</td>
</tr>
<tr>
<td>Read for Copy Back</td>
<td>00h</td>
<td>35h</td>
</tr>
<tr>
<td>Read ID</td>
<td>90h</td>
<td>-</td>
</tr>
<tr>
<td>Reset</td>
<td>FFh</td>
<td>-</td>
</tr>
<tr>
<td>Page Program</td>
<td>80h</td>
<td>10h</td>
</tr>
<tr>
<td>Cache Program</td>
<td>80h</td>
<td>15h</td>
</tr>
<tr>
<td>Copy-Back Program</td>
<td>85h</td>
<td>10h</td>
</tr>
<tr>
<td>Block Erase</td>
<td>60h</td>
<td>00h</td>
</tr>
<tr>
<td>Random Data Input*</td>
<td>85h</td>
<td>-</td>
</tr>
<tr>
<td>Random Data Output*</td>
<td>05h</td>
<td>E0h</td>
</tr>
<tr>
<td>Read Status</td>
<td>70h</td>
<td></td>
</tr>
</tbody>
</table>

*Samsung K9K4G08U0M datasheet

**Table 4: Command Set**

<table>
<thead>
<tr>
<th>FUNCTION</th>
<th>1st CYCLE</th>
<th>2nd CYCLE</th>
</tr>
</thead>
<tbody>
<tr>
<td>READ 1</td>
<td>00h</td>
<td>30h</td>
</tr>
<tr>
<td>READ FOR COPY-BACK</td>
<td>00h</td>
<td>35h</td>
</tr>
<tr>
<td>READ ID</td>
<td>90h</td>
<td>-</td>
</tr>
<tr>
<td>RESET</td>
<td>FFh</td>
<td>-</td>
</tr>
<tr>
<td>PAGE PROGRAM (start)</td>
<td>80h</td>
<td>10h</td>
</tr>
<tr>
<td>COPY BACK PGM (start)</td>
<td>85h</td>
<td>10h</td>
</tr>
<tr>
<td>CACHE PROGRAM</td>
<td>80h</td>
<td>15h</td>
</tr>
<tr>
<td>BLOCK ERASE</td>
<td>60h</td>
<td>D0h</td>
</tr>
<tr>
<td>READ STATUS REGISTER</td>
<td>70h</td>
<td>-</td>
</tr>
<tr>
<td>RANDOM DATA INPUT</td>
<td>85h</td>
<td>-</td>
</tr>
<tr>
<td>RANDOM DATA OUTPUT</td>
<td>05h</td>
<td>E0h</td>
</tr>
<tr>
<td>CACHE READ START</td>
<td>00h</td>
<td>31h</td>
</tr>
<tr>
<td>CACHE READ EXIT</td>
<td>34h</td>
<td>-</td>
</tr>
<tr>
<td>LOCK BLOCK</td>
<td>2Ah</td>
<td>-</td>
</tr>
<tr>
<td>LOCK TIGHT</td>
<td>2Ch</td>
<td>-</td>
</tr>
<tr>
<td>UNLOCK (start area)</td>
<td>23h</td>
<td>-</td>
</tr>
<tr>
<td>UNLOCK (end area)</td>
<td>24h</td>
<td>-</td>
</tr>
<tr>
<td>READ LOCK STATUS</td>
<td>7Ah</td>
<td>-</td>
</tr>
</tbody>
</table>

*Hynix HY27UG084G2M datasheet

*Other names and brands may be claimed as the property of others*
System problems with NAND Today

**Similar: Timing**

Timing requirements are specified differently
Min/max values are not necessarily the same for similar cycle time parts

---

*Samsung K9K4G08U0M datasheet

*Hynix HY27UG084G2M datasheet

*Other names and brands may be claimed as the property of others
System problems with NAND Today

**Similar: Status Values**

Status values dependent on command

Often the same, but not required to be

–Can you tell that these are the same??

<table>
<thead>
<tr>
<th>I/O No.</th>
<th>Page Program</th>
<th>Block Erase</th>
<th>Cache Program</th>
<th>Read</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>I/O 0</td>
<td>Pass/Fail</td>
<td>Pass/Fail</td>
<td>Pass/Fail(N)</td>
<td>Not use</td>
<td>Pass: '0' Fail: '1'</td>
</tr>
<tr>
<td>I/O 1</td>
<td>Not use</td>
<td>Not use</td>
<td>Pass/Fail(N-1)</td>
<td>Not use</td>
<td>Pass: '0' Fail: '1'</td>
</tr>
<tr>
<td>I/O 2</td>
<td>Not use</td>
<td>Not use</td>
<td>Not use</td>
<td>Not use</td>
<td>Don't care</td>
</tr>
<tr>
<td>I/O 3</td>
<td>Not Use</td>
<td>Not Use</td>
<td>Not Use</td>
<td>Not Use</td>
<td>Don't care</td>
</tr>
<tr>
<td>I/O 4</td>
<td>Not Use</td>
<td>Not Use</td>
<td>Not Use</td>
<td>Not Use</td>
<td>Don't care</td>
</tr>
<tr>
<td>I/O 5</td>
<td>Ready/Busy</td>
<td>Ready/Busy</td>
<td>True Ready/Busy</td>
<td>Ready/Busy</td>
<td>Busy: '0' Ready: '1'</td>
</tr>
<tr>
<td>I/O 6</td>
<td>Ready/Busy</td>
<td>Ready/Busy</td>
<td>True Ready/Busy</td>
<td>Ready/Busy</td>
<td>Busy: '0' Ready: '1'</td>
</tr>
<tr>
<td>I/O 7</td>
<td>Write Protect</td>
<td>Write Protect</td>
<td>Write Protect</td>
<td>Write Protect</td>
<td>Protected: '0' Not Protected: '1'</td>
</tr>
</tbody>
</table>

*Hynix HY27UG084G2M datasheet

*Samsung K9K4G08U0M datasheet

*Other names and brands may be claimed as the property of others
System problems with NAND Today

**Similar: Read ID**

The first and second byte are consistently manufacturer and device ID

The number of remaining bytes and what they mean is up in the air

*Samsung K9K4G08U0M datasheet

<table>
<thead>
<tr>
<th>1st Byte</th>
<th>Description</th>
<th>I/O8</th>
<th>I/O7</th>
<th>I/O6</th>
<th>I/O5</th>
<th>I/O4</th>
<th>I/O3</th>
<th>I/O2</th>
<th>I/O1</th>
<th>Hex Data</th>
</tr>
</thead>
<tbody>
<tr>
<td>Maker Code</td>
<td>Device Code</td>
<td>0000</td>
<td>1001</td>
<td>1001</td>
<td>1001</td>
<td>1001</td>
<td>1001</td>
<td>1001</td>
<td>1001</td>
<td>98H</td>
</tr>
<tr>
<td>Device Code</td>
<td>Device Code</td>
<td>0000</td>
<td>1001</td>
<td>1001</td>
<td>1001</td>
<td>1001</td>
<td>1001</td>
<td>1001</td>
<td>1001</td>
<td>DAH</td>
</tr>
<tr>
<td>Chip Number, Cell Type, PGM Page, Write Cache</td>
<td>Chip Number, Cell Type, PGM Page, Write Cache</td>
<td>0000</td>
<td>1001</td>
<td>1001</td>
<td>1001</td>
<td>1001</td>
<td>1001</td>
<td>1001</td>
<td>1001</td>
<td>81H or 01H</td>
</tr>
<tr>
<td>Page Size, Block Size, Redundant Size, Organization</td>
<td>Page Size, Block Size, Redundant Size, Organization</td>
<td>0000</td>
<td>1001</td>
<td>1001</td>
<td>1001</td>
<td>1001</td>
<td>1001</td>
<td>1001</td>
<td>1001</td>
<td>95H or 15H</td>
</tr>
<tr>
<td>Plane Number, Plane Size</td>
<td>Plane Number, Plane Size</td>
<td>0000</td>
<td>1001</td>
<td>1001</td>
<td>1001</td>
<td>1001</td>
<td>1001</td>
<td>1001</td>
<td>1001</td>
<td>44H or C4H</td>
</tr>
</tbody>
</table>

*Toshiba TH58NVG1S3AFT05 datasheet

*Other names and brands may be claimed as the property of others
Similar Hinders NAND Adoption

To deal with differences, the host must maintain a chip ID table of known devices
- Table contains read/write timings, organization, status bit meanings, etc for each known NAND Flash part

Situation has two major effects:
- Precludes intro of new NAND devices into existing designs
- Makes qualification cycles longer as each NAND device added requires changes to be comprehended

Similar to the ancient disk drive interfaces that required a list of disk drive types in a BIOS table

Lack of standard impacts platforms supporting a range of NAND
Agenda

NAND Flash Product Integration
System problems with NAND Today
Open NAND Flash Interface (ONFI) initiative overview
ONFI Technical Preview
Open NAND Flash Interface

Goals of Initiative:

To develop a standardized NAND Flash interface that allows interoperability between NAND devices

Accelerate time to market of NAND-based products
Technical Philosophy

ONFI shall ensure no pre-association with NAND Flash at host design is required
- Flash must self-describe features, capabilities, timings, etc, through a parameter page
- Features that cannot be self-described in a parameter page (like number of CE#) shall be host discoverable

ONFI should leverage existing Flash behavior to the extent possible
- Intent is to enable orderly and TTM transition, so highly divergent behavior from existing NAND undesired
- Where prudent for longevity or capability need, existing Flash behavior shall be modified or expanded

ONFI needs to enable future innovation
Initiative Status

Intel is working with key partners to form the ONFI Workgroup

– More details to come shortly

ONFI specification expected to be released in 2H’06

ONFI is being developed to solve the barriers to the rapid adoption of new NAND products
Agenda

NAND Flash Product Integration
System problems with NAND Today
Open NAND Flash Interface (ONFI) initiative overview

ONFI Technical Preview
ONFI Technical Preview

Pin-Out and Package

ONFI defines standard pin-outs for the 48-pin TSOP and 52-pin LGA packages

Pin-out critical for ensuring no board changes required in NAND Flash upgrade

- For example, the host can plan for a new part that will have an additional CE# beforehand

![Diagram showing pin-Out and Package for the 48-pin TSOP and 52-pin LGA packages]
Fixing 16-bit Data Pin-out

The current industry x16 pin-out precludes board designs accepting both x8 and x16 parts easily

- The I/O0 – I/O7 pins are not in the same place!

ONFI is fixing this issue before x16 parts become entrenched (x16 currently is ~ < 1% of volume)

![Industry x16 pin-out diagram](image)

6 of 8 data pins are in different locations on the x16 pin-out!
Device Abstraction

ONFI presents a device representation to the host based on independence “levels”

– ONFI has no notion of number of die or number of planes, etc.

Target: Completely independent unit with its own chip enable (CEx#)

– Logical Unit (LUN): Logically independent unit, shared chip enable

– Interleaved Addressing: Dependent operations allowed for different blocks with same page address
Targets and LUNs

Each target may support multiple logical units
Each LUN may support interleaved addressing for increased parallelism
Actual implementation abstracted
Determining ONFI Support

Read ID is used by Flash parts today to report device ID for use in chip ID table lookup.

ONFI support is shown by responding to Read ID for address 20h with ASCII ‘ONFI’.

Support for vendor specific interface and ONFI allowed by changing address cycle to 20h from 0h.

ONFI Technical Preview

Intel Developer FORUM

28
Self Reporting Capabilities

Each target describes its features and capabilities through a parameter page. Blocks of the parameter page are devoted to:

- Revision information
- Device features
- Manufacturer information
- Memory organization
- Timing parameters
- Vendor specific

Memory Organization Block of Parameter Page

- Number of data bytes per page
- Number of redundant bytes per page
- Number of pages per block
- Number of blocks per logical unit (LUN)
- Number of logical units
- Number of address cycles
- Number of bits per cell
- Block endurance
- Number of programs per page
- Recommended bits of ECC correction
- Interleaved addressing
## Command Set Overview

<table>
<thead>
<tr>
<th>Command</th>
<th>O/M</th>
<th>1&lt;sup&gt;st&lt;/sup&gt; Cycle</th>
<th>2&lt;sup&gt;nd&lt;/sup&gt; Cycle</th>
<th>Acceptable while Accessed LUN is Busy</th>
<th>Acceptable while Other LUNs are Busy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Read</td>
<td>M</td>
<td>00h</td>
<td>30h</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>Change Read Column</td>
<td>M</td>
<td>05h</td>
<td>E0h</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>Change Read Column Enhanced</td>
<td>O</td>
<td>06h</td>
<td>E0h</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>Read Cache</td>
<td>O</td>
<td>31h</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Read Cache End</td>
<td>O</td>
<td>3Fh</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Block Erase</td>
<td>M</td>
<td>60h</td>
<td>D0h</td>
<td></td>
<td>Y</td>
</tr>
<tr>
<td>Reserved</td>
<td></td>
<td>61h – 6Fh</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Read Status</td>
<td>M</td>
<td>70h</td>
<td></td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>Read LUN Status</td>
<td>O</td>
<td>78h</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Page Program</td>
<td>M</td>
<td>80h</td>
<td>10h</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Page Cache Program</td>
<td>O</td>
<td>80h</td>
<td>15h</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Change Write Column</td>
<td>M</td>
<td>85h</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Read ID</td>
<td>M</td>
<td>90h</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Vendor Unique</td>
<td></td>
<td>91h - BFh</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Read Parameter Page</td>
<td>M</td>
<td>ECh</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Get Features</td>
<td>O</td>
<td>EEh</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Set Features</td>
<td>O</td>
<td>EFh</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Reset</td>
<td>M</td>
<td>FFh</td>
<td></td>
<td>Y</td>
<td>Y</td>
</tr>
</tbody>
</table>
Timing Requirements

NAND contains a lot of timing requirements and hence timings

Reporting each and every timing value leads to validation challenge

- Requires validation of all combinations

To make timing information useful, organized into timing modes

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>tADL</td>
<td>Minimum ALE to data loading time</td>
</tr>
<tr>
<td>tALH</td>
<td>Minimum ALE hold time</td>
</tr>
<tr>
<td>tALS</td>
<td>Minimum ALE setup time</td>
</tr>
<tr>
<td>tAR</td>
<td>Minimum ALE to RE# delay</td>
</tr>
<tr>
<td>tBERS</td>
<td>Maximum block erase time</td>
</tr>
<tr>
<td>tCEA</td>
<td>Maximum CE# access time</td>
</tr>
<tr>
<td>tCH</td>
<td>Minimum CE# hold time</td>
</tr>
<tr>
<td>tCHZ</td>
<td>Maximum CE# high to output hi-Z</td>
</tr>
<tr>
<td>tCLOAD</td>
<td>Minimum CLE to RE# delay</td>
</tr>
<tr>
<td>tRLOH</td>
<td>Minimum RE# low to output hold</td>
</tr>
<tr>
<td>tRP</td>
<td>Minimum RE# pulse width</td>
</tr>
<tr>
<td>tRR</td>
<td>Minimum Ready to RE# low</td>
</tr>
<tr>
<td>tRST</td>
<td>Maximum device reset time</td>
</tr>
<tr>
<td>tWB</td>
<td>Maximum WE# high to R/B# low</td>
</tr>
<tr>
<td>tWC</td>
<td>Minimum write cycle time</td>
</tr>
<tr>
<td>tWH</td>
<td>Minimum WE# high hold time</td>
</tr>
<tr>
<td>tWHR</td>
<td>Minimum WE# high to RE# low</td>
</tr>
<tr>
<td>tWP</td>
<td>Minimum WE# pulse width</td>
</tr>
</tbody>
</table>
Timing Modes

Timing modes define vast majority of required host timings as one “set”

Three parameters are specified separately in RPP

- Max page read time
- Max block erase time
- Max page program time

Timing modes supported reported in timing parameters block of RPP

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Mode Example</th>
<th>Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>tADL</td>
<td>100</td>
<td>ns</td>
</tr>
<tr>
<td>tAH</td>
<td>5</td>
<td>ns</td>
</tr>
<tr>
<td>tALS</td>
<td>10</td>
<td>ns</td>
</tr>
<tr>
<td>tAR</td>
<td>10</td>
<td>ns</td>
</tr>
<tr>
<td>tCEA</td>
<td>25</td>
<td>ns</td>
</tr>
<tr>
<td>tCH</td>
<td>5</td>
<td>ns</td>
</tr>
<tr>
<td>tCHZ</td>
<td>20</td>
<td>ns</td>
</tr>
<tr>
<td>tCLH</td>
<td>5</td>
<td>ns</td>
</tr>
<tr>
<td>tCLR</td>
<td>10</td>
<td>ns</td>
</tr>
<tr>
<td>tCLS</td>
<td>15</td>
<td>ns</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>ns</td>
</tr>
<tr>
<td>tWB</td>
<td>100</td>
<td>ns</td>
</tr>
<tr>
<td>tWC</td>
<td>30</td>
<td>ns</td>
</tr>
<tr>
<td>tWH</td>
<td>10</td>
<td>ns</td>
</tr>
<tr>
<td>tWHR</td>
<td>60</td>
<td>ns</td>
</tr>
<tr>
<td>tWP</td>
<td>15</td>
<td>ns</td>
</tr>
</tbody>
</table>
Interleaved Operation

Interleaving may be used to complete the same operation on additional blocks on a per logical unit basis to enhance performance.

- Concurrent interleaving: Operations to all of the blocks is issued at the same time and then executes in parallel.
- Overlapped interleaving: Operations may be issued independently, allows host to determine later to do an additional operation.

The operations that may be interleaved are reads, programs, and erases.

When using interleaving, the lowest order bits of the block address may be modified. The rest of the address must be the same as the other operations being issued on that LUN.
Defect Mapping & Enumeration

An invalid block is indicated by a 00h value in the first or last page of a block

- Ensures robustness in marking bad pages in face of recoverable bit errors

The host uses this information to create its initial bad block table

```c
// For each LUN maps defects
for (i = 0; i < NumLUNs; i++) {

    // For each block within this LUN, map defects
    for (j = 0; j < BlocksPerLUN; j++) {
        Defective=FALSE;

        // If a 00h value is in the first page, this block is defective
        ReadPage(lun=i; block=j; page=0; DestBuff=Buff);
        for (column=0; column<PageSize+RedundantBytes; column++) {
            if (Buff[column] == 00h)
                Defective=TRUE;
        }

        // If a 00h value is in the last page, this block is defective
        ReadPage(lun=i; block=j; page=PagesPerBlock-1; DestBuff=Buff);
        for (column = 0; column < PageSize+RedundantBytes; column++) {
            if (Buff[column] == 00h)
                Defective=TRUE;
        }

        // If the block was defective, then keep track of this
        if (Defective)
            MarkBlockDefective(lun=i; block=j);
    }
}
```
ONFI Technical Highlights

ONFI support is identified via Read ID, the standard NAND chip ID command.
NAND devices report their capabilities using the Read Parameter Page.
ONFI standardizes the base subset of commands required to be supported by all NAND devices.
ONFI supports increased performance through parallelism made possible by multiple LUNs and interleaved addressing.
ONFI standardizes the pin-out and packaging to ensure no PCB changes are required for a new NAND part.

ONFI provides the solid technical base necessary to confidently build NAND-based products.
Summary

Differing implementations of the NAND Flash interface across vendors impacts time to market and revenue.

Lack of standard makes it impossible for platforms to support a range of NAND components, including components introduced at a later date.

ONFI is being developed to solve the barriers to the rapid adoption of new NAND products.

ONFI enables NAND feature self-identification, and standardizes command set, pin-out, and packaging.
Please fill out the Session Evaluation Form.

Additional sources of information on this topic:
IDF Chalk Talk, MEMC007: Upcoming Flash and Main Memory Technologies and Applications
Initiative updates available at: www.onfi.org

Session presentation available on IDF web site –
when prompted enter:
Username: idf
Password: SPR2006

Thank You!
Backup Slides