

# Technical Note

## Memory Management in NAND Flash Arrays

---

### Overview

NAND Flash devices have established a strong foothold in solid-state mass storage, as both a removable storage medium and an embedded storage medium. As NAND Flash adoption has increased, the variety of configurations in which NAND Flash devices are employed has also increased. This proliferation is due in part to the various ways NAND Flash memory arrays can be combined.

The way NAND Flash memory arrays are combined can have a major impact on application performance; it also influences the resources that will be required to organize and maintain a NAND Flash memory array.

This technical note describes common NAND Flash memory-management methods, advantages and considerations associated with each, and suggestions for the most effective use of the NAND Flash memory array. A solid understanding of NAND Flash devices is necessary for designers incorporating the concepts discussed in this document.

The examples used are based on the Micron<sup>®</sup> MT29F4G08AAA NAND Flash device. For detailed information on this device, refer to the full data sheet at [www.micron.com/products/nand](http://www.micron.com/products/nand).

## Performance Benefits of Combining NAND Flash Memory Arrays

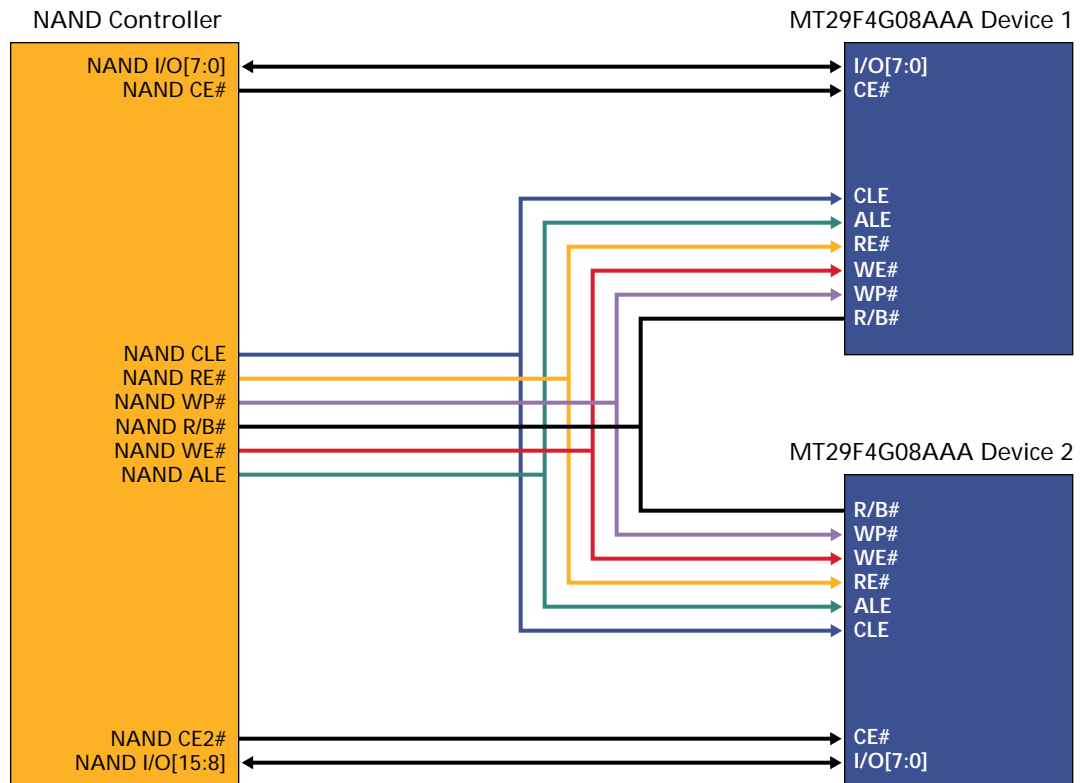
Combining NAND Flash memory arrays can improve performance by increasing the apparent size of the NAND Flash data register and the apparent size of individual blocks. Parallel configurations support a higher volume of bytes being read, programmed, or erased simultaneously.

### Benefits of Combining NAND Blocks across Chip Enables (CE#s)

The asynchronous nature and the block memory structure of NAND Flash devices support combining NAND Flash devices across chip enables (CE#s) in a relatively straightforward process (see Figure 1).

Shorting CE#s together is a common practice when combining NAND Flash blocks. By not shorting CE#s together, as shown in Figure 1, a design retains the flexibility to communicate with only one NAND device at a time.

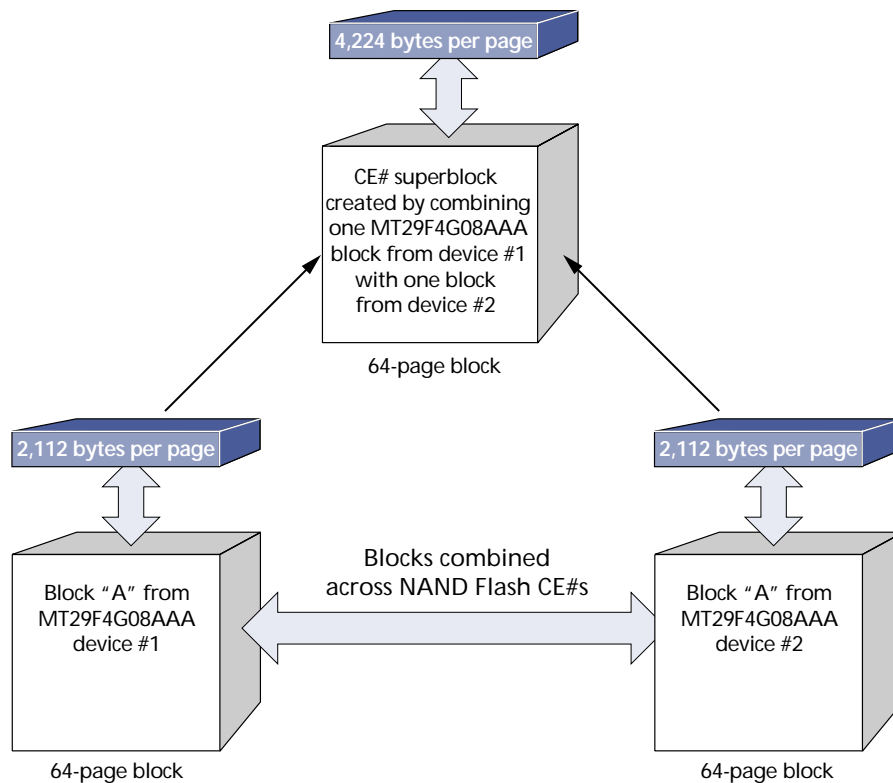
Figure 1: NAND Flash Blocks Combined Across CE#s



### Creating Superblocks

A “CE# superblock” uses shared command signals (CLE, ALE, WE#, RE#, WP#, R/B#), with input/output signals (I/O[7:0]) and CE# signals distributed among two or more NAND Flash memory arrays on two or more CE#s. This configuration makes it possible to combine two or more NAND Flash blocks across CE#s to form the CE# superblock. Figure 1 depicts an implementation using two MT29F4G08AAA NAND devices with blocks combined across CE#s. Figure 2 shows the creation of a CE# superblock using two NAND Flash blocks.

Figure 2: NAND Flash Blocks Combined to Create a CE# Superblock



### Superblock Shared-Signal Benefits

Each NAND Flash memory array sharing command signals receives the same command instructions when its own CE# is active (LOW). For example, when a PROGRAM PAGE (80h-10h) command is sent on the shared command signals, each NAND Flash memory array will respond and execute a PROGRAM PAGE (80h-10h) operation.

If the I/O[7:0] and CE# signals are not shared among multiple NAND Flash memory arrays, each NAND Flash memory array can address different addresses and different data when implementing operations. This enables selection of a single NAND Flash memory array for specific operations when it is advantageous to do so.

### Superblock Configuration

A CE# superblock is generally comprised of a shared CE# and the same physical block from each NAND Flash memory used. Different physical blocks in the NAND Flash memory arrays can be used; however, using the same physical block in each NAND Flash device simplifies block management.

The system design designates which physical blocks in each array are combined as a CE# superblock. When the superblock is accessed, the combined blocks on each CE# are being accessed at the same time, providing parallel access to the number of bytes in an MT29F4G08AAA page (2,112 bytes) multiplied by the number of combined arrays sharing the CE#.

Combining across the CE#s makes it possible to execute the same command on all of the combined arrays in parallel, and to input data, read data, or erase data on all of the combined arrays in parallel.

The superblock enables a significantly higher volume of data movement across the combined arrays, with  $X$  times the number of bytes programmed or read from the array for each  $t^{\text{WC}}$  or  $t^{\text{RC}}$  issued from the NAND controller, using only the  $t^{\text{PROG}}$  or  $t^{\text{R}}$  array access time associated with single program or read operations.

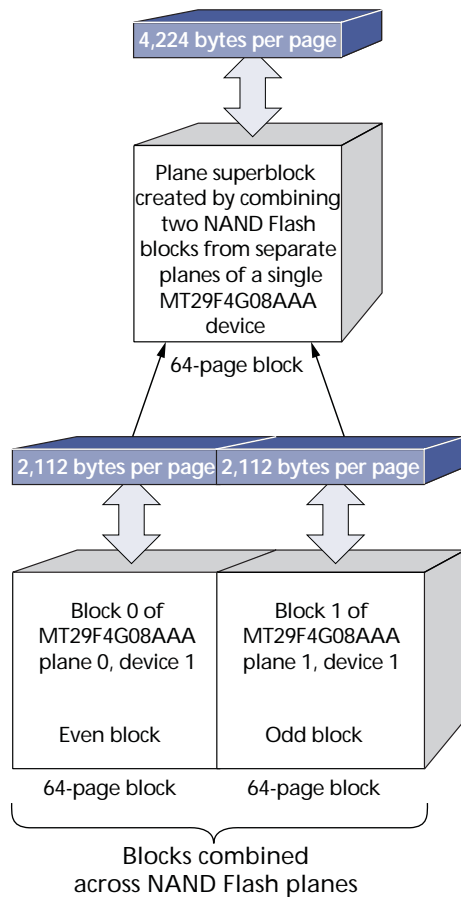
The superblock also enables improved erase performance by erasing two blocks, one on each CE# by way of the combined arrays, within the  $t^{\text{BERS}}$  time associated with a single BLOCK ERASE operation.

### Benefits of Combining NAND Blocks across NAND Planes

Similar to combining NAND blocks across CE#s, combining NAND blocks across NAND planes creates a “plane superblock” made up of two or more NAND Flash blocks. The difference is that when combining NAND Flash blocks across planes, the combined blocks exist on the same NAND Flash die. The plane superblock has a page size of  $X$  bytes, where  $X$  is the number of NAND planes combined, multiplied by the number of bytes per page.

The difference between CE# superblocks and plane superblocks is that the combined blocks of a plane superblock exist on the same CE#. Figure 3 on page 5 depicts creation of a plane superblock using two NAND Flash blocks that reside on different planes of the same CE#.

Figure 3: NAND Flash Blocks Combined to Create a Plane Superblock



A plane superblock is generally made up of adjacent physical blocks, one from each plane (e.g., block 0 with block 1, or block 2 with block 3). Micron NAND Flash devices have the ability to use adjacent physical blocks or nonadjacent physical blocks that are on different planes to create plane superblocks. The system design determines which physical blocks in each NAND Flash plane are combined as a plane superblock.

The performance improvement using plane superblocks is not as significant as the performance improvement realized using CE# superblocks. Independent data signals going to the NAND Flash memory arrays combined between CE#s deliver parallel command, address, or data transmissions. Commands, addresses, and data are transmitted serially to each NAND Flash plane in a plane superblock, requiring more time to complete all of the transactions; the second-plane data input also requires additional I/O time.

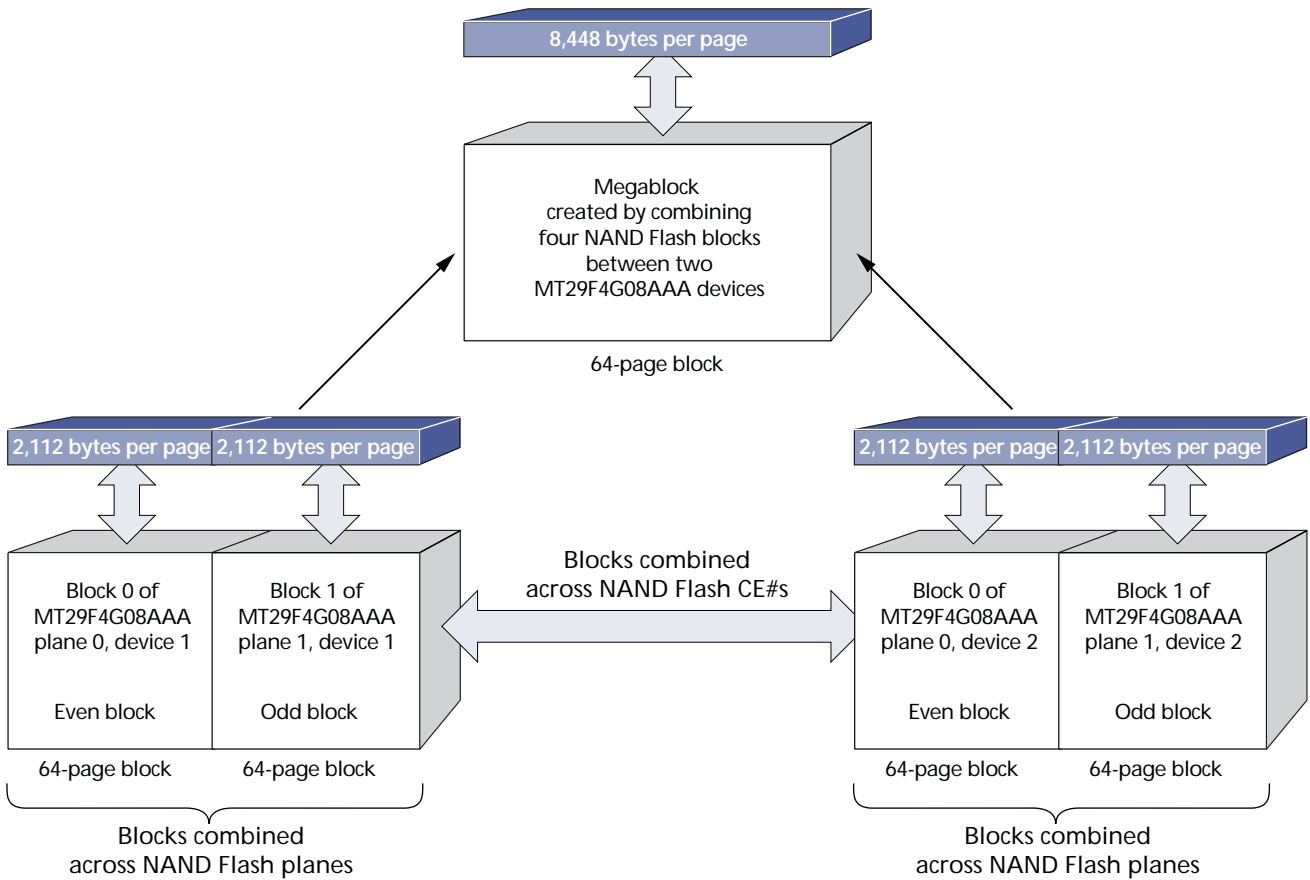
### Achieving Optimal Performance

To obtain the best performance in a linked NAND Flash configuration, use Micron two-plane commands. These commands are described in detail in the MT29F4G08AAA data sheet at [www.micron.com/products/nand](http://www.micron.com/products/nand). Supplementary information regarding two-plane commands is available in Technical Note TN-29-25, "Improved Performance Using Two-Plane Commands," at [www.micron.com/products/nand/technotes](http://www.micron.com/products/nand/technotes).

### Benefits of Combining NAND Blocks across CE#s and NAND Planes

Combining NAND Flash blocks across both CE#s and NAND planes creates a “megablock” made up of four or more NAND Flash blocks. Two sets of two-plane commands can be issued simultaneously to a megablock using two independent data channels. Creating a megablock delivers the greatest possible throughput in Micron NAND Flash devices. Figure 4 shows the creation of a megablock using four NAND Flash blocks from two different NAND Flash devices.

**Figure 4: NAND Flash Blocks Linked to Create a Megablock**



## Linking Trade-offs

As with most memory architecture options, there are trade-offs when organizing NAND Flash memory arrays by linking NAND blocks across CE#s and/or NAND planes. When using the most common approach—linking the same physical blocks together across CE#s and linking physically adjacent NAND blocks together across NAND planes—the potential trade-off is reduced NAND Flash storage.

This common approach makes it possible for the same cluster of blocks in each linked arrangement to always be linked together, and facilitates easy file management and memory maintenance. If any of the individual blocks are bad, for simplicity the file management software will assume that any block combined with a bad block is unusable.

There are some drawbacks to these linking arrangements. Linking NAND blocks together without consideration for bad blocks can cause a reduction in available storage capacity. If just one of the combined blocks is bad, then all the blocks combined with that bad block are also considered bad. This can lead to a situation where, in an effort to realize the benefits of linking noted previously, good blocks end up being designated as bad, thus reducing storage availability. Each designer must weigh the benefits of linking against the potential for reduced storage.

By combining NAND Flash blocks blindly across CE#s and/or across NAND planes without first considering any bad blocks that may be present, the total number of bad blocks in a system increases artificially; the good blocks combined with bad blocks can double the total bad-block count. This creates a situation where a NAND Flash device that meets its data sheet specification for the number of bad blocks may not meet the minimum number of valid blocks required by a system due to the artificially increased bad-block count.

This potential artificial memory loss will increase as more blocks are combined, particularly if future applications use greater numbers of NAND Flash devices or devices with more than two planes.

## Effective Linking Methods

Designers can minimize the number of good blocks artificially made bad by being linked to bad blocks. The linking scenarios that follow demonstrate the differences between linking methods, and the best approach for conserving the greatest number of good blocks.

### Block Preservation Linking Across CE#s

There is little chance that a bad physical block on one device's CE# will also be bad on the CE# of another linked device. This being the case, linking CE#s without checking the viability of the blocks being linked can lead to bad-block doubling when a bad block is linked unknowingly with a good block. Bad-block doubling automatically reduces the number of valid blocks and, thus, the available density.

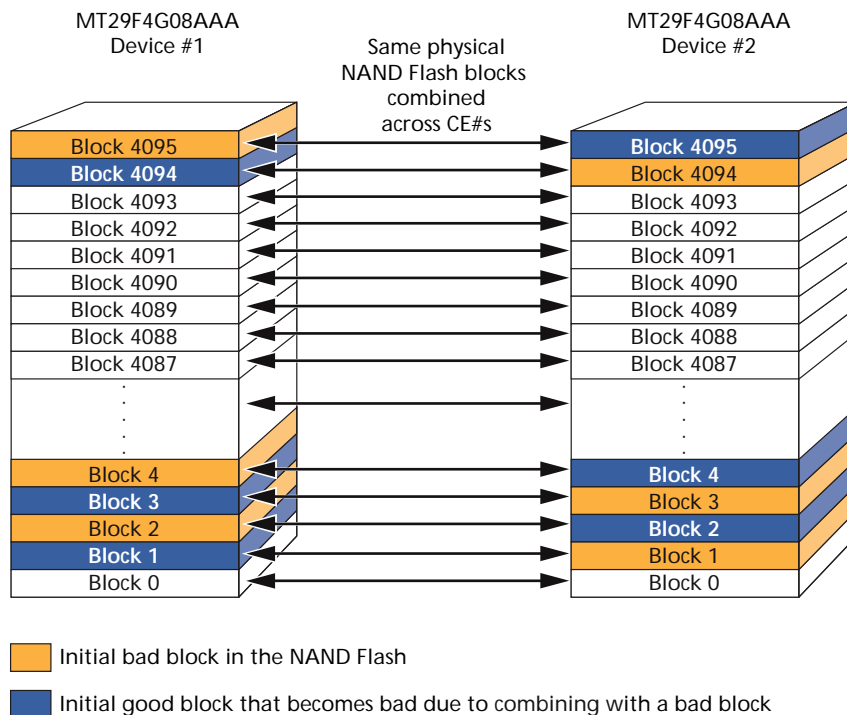
Designers can use linking to their advantage by first checking for physical bad blocks in a multiple NAND-CE# environment, then linking different identified bad blocks from each device on the CE#s. This method improves throughput and preserves available density without artificially increasing bad-block overhead.

### Same-Block Linking

For example, consider two NAND Flash devices with the same physical blocks in each device linked together across CE#s (see Figure 5). Each of the devices has only three bad blocks, and the bad blocks in one device are at different physical block locations than the bad blocks in the other device.

When the same physical blocks are linked across the CE#s of the two NAND devices, and each of the bad blocks is at a unique physical block location, the total number of bad blocks between the two NAND devices doubles from six to twelve.

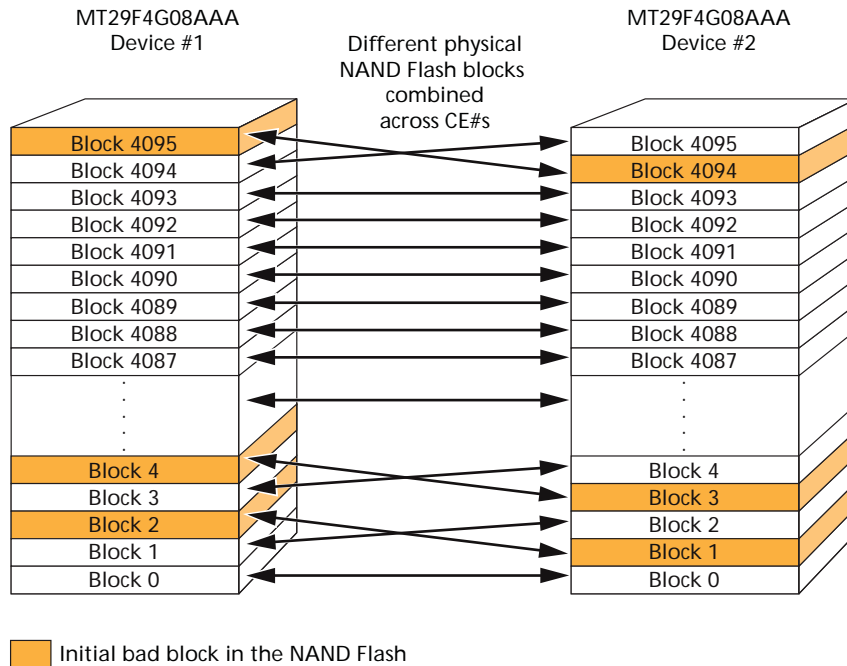
Figure 5: Linking Using the Same Physical Blocks



### Different-Block Linking

Now consider the same two NAND Flash devices with three bad blocks in each device, and different bad-block locations in each device (see Figure 6). By combining the bad blocks together on the CE#s, rather than combining the same physical block location on each device, only bad blocks are combined; no good blocks are wasted, and the total bad-block count between the two NAND Flash devices remains at six.

**Figure 6: Linking Using Different Physical Blocks**



### Block Preservation Linking Across Planes

The concept presented for combining across CE#s also applies when linking NAND Flash blocks across planes within a die. When adjacent physical blocks are combined arbitrarily across the planes of a die, the potential for linking good blocks to bad increases.

By verifying block validity and linking bad blocks across planes, rather than automatically linking adjacent blocks, the total bad-block count is retained, no good blocks are wasted, and available density is preserved.

### Optimized Linking

Managing block linking makes it possible to preserve available device density by preventing artificial bad-block multiplication. The process is somewhat more complex than simply combining the same blocks in each device via CE#, or adjacent blocks across planes. It requires dynamically keeping track of blocks between CE#s and/or planes over the lifetime of the NAND Flash devices.

The payoff for managing block linking up front includes reduced bad-block overhead, maximum available density, and less chance that the application will fall below minimum density requirements over the operational life of the device.

## Summary

Performance in NAND Flash solid-state storage can be improved by using an architectural and system-level approach to create various types of linked superblocks. It is important to consider the side effects that may be imposed on a system when linking NAND Flash blocks together. The actual effects will depend on how block linking is managed. Each of the two possible linking methods—linking based on physical block location, or dynamic linking—are accompanied by inherent advantages and disadvantages.

The easiest approach to implementing and maintaining a linked system is linking NAND Flash blocks together based strictly on physical location in the NAND device, with no regard for any bad blocks that may already exist in the system. This approach is accompanied by risks associated with the high potential for reduced available NAND Flash density over the operational life of the system, as blocks go bad over time.

A more efficient approach to linked systems is dynamic linking accomplished by initially pairing blocks that are already bad, and later pairing any that go bad over time. This approach is more difficult to implement and maintain, but it helps retain more valid-block density and reduces the risk of falling below minimum density requirements, as blocks go bad over time.

The latest information on Micron NAND Flash devices is available at [www.micron.com/products/nand/](http://www.micron.com/products/nand/).



8000 S. Federal Way, P.O. Box 6, Boise, ID 83707-0006, Tel: 208-368-3900  
prodmktg@micron.com www.micron.com Customer Comment Line: 800-932-4992  
Micron, the M logo, and the Micron logo are trademarks of Micron Technology, Inc.  
All other trademarks are the property of their respective owners.