Migrating Between Boot & Uniform Sectored Flash Devices

About this document

Scope and purpose

AN99113 describes the software algorithm changes necessary to convert from a boot sectored flash device to a uniform sectored Flash device or vice versa.

Table of contents

1 Overview

The purpose of this application note is to describe the software algorithm changes necessary to convert from a boot sectored Flash device to a uniform sectored Flash device or vice versa. Changes are required only for applications that require sector erase capabilities in a system. If sector erase operation is not performed within a system, there is no functional difference between boot and uniform sectored devices and therefore no need to make any software changes.

2 The Origins of Sector Erase Flash Memories

The first generation Flash devices manufactured were bulk erase Flash devices. Bulk erase Flash devices could program (change a “1” to a “0”) information into the device a byte at one time, but required the entire Flash device to be erased (change “0's” to “1's”) every time a byte of memory needed to be erased. This wasn't the most efficient way of changing data because you had to erase and reprogram the whole Flash device for any erase no matter how small the change. To solve this problem, Flash devices with erasable sectors were developed. Now, instead of erasing the entire Flash device at one time, you could erase all memory locations within a sector independently. Since Flash devices with sector erase were introduced, it has further developed into various organizations to better suit the needs of today's applications. Flash devices now are organized into two different architectures, boot or uniform sectored. Sectors now also have the ability to be individually or group protected against accidental programming or erasure.

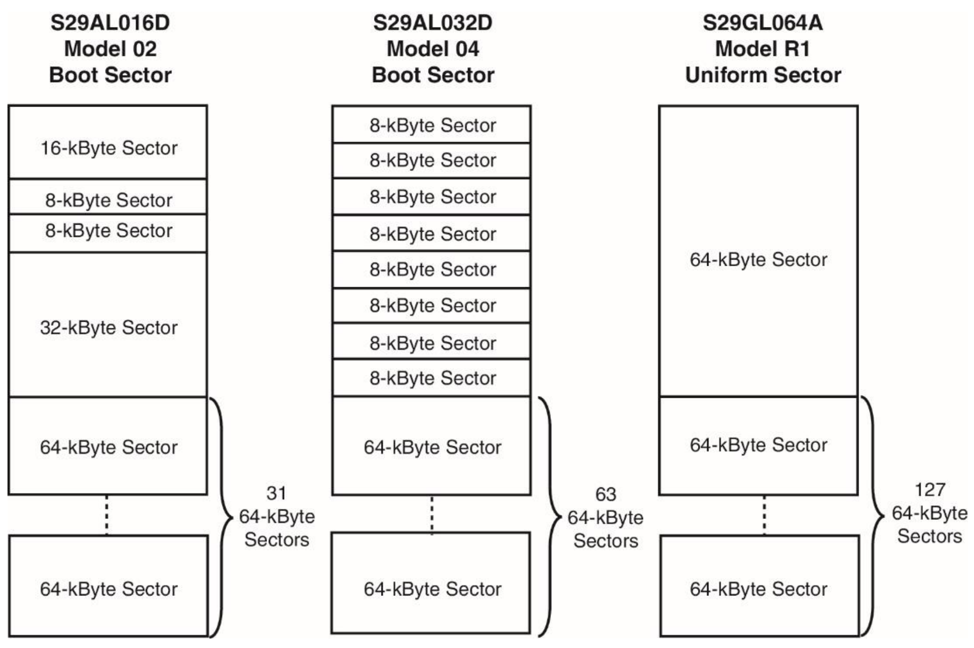
3 The Difference Between Boot and Uniform Sectored Flash Architectures

The difference between a boot sectored device and a uniform sectored device is the make up of the first or last 64 Kbytes of memory. For a uniform sectored device, the 64 Kbytes are a single erase block, which is similar to the rest of the sectors within the device. Uniform sectored devices are ideal for building large memory arrays since the blocks are all the same size and therefore easily managed. For boot devices, the first or last 64-Kbyte sector is subdivided into several smaller sectors. Two typical configurations of these smaller sectors are 1-16 Kbytes, 2-8 Kbytes, 1-32 Kbytes and 8-8 Kbyte sectors.

These smaller sectors are generally used to store boot code or parametric data. The reason for this is that boot code and parametric data usually do not require a whole 64-Kbyte sector, but still need to be placed in a separate sector so it can be properly protected or changed without affecting other portions of code. Smaller boot sectors permit this without having to waste an entire 64-Kbyte sector. Boot sectored devices also have the option of placing the smaller boot sectors within the first 64 Kbytes of the low address range (called bottom boot) or the last 64 Kbytes of the high address range (called top boot). This is due to the way in which microcontrollers access the boot code after power-on. Two conventions for microcontrollers exist:

1. Top-boot convention, where the first address fetch after power-on is performed at the top of the memory space (FF...FF0h). Microcontrollers based on AMD® and Intel® X86 architecture processors boot from the top of the memory space.
2. Bottom-boot convention, where the first address fetch is performed at the bottom of the memory space (00...000h). Microcontrollers based on IBM®, RISC, and Motorola® processors boot from the bottom of the memory space.

The following are examples of low voltage 16-Mbit and 32-Mbit boot sectored devices as well as a 64-Mbit uniform sectored device.



4 Software Changes When Migrating From Boot To Uniform

4.1 Case 1

The system allows all boot sectors to be erased together.

Solution

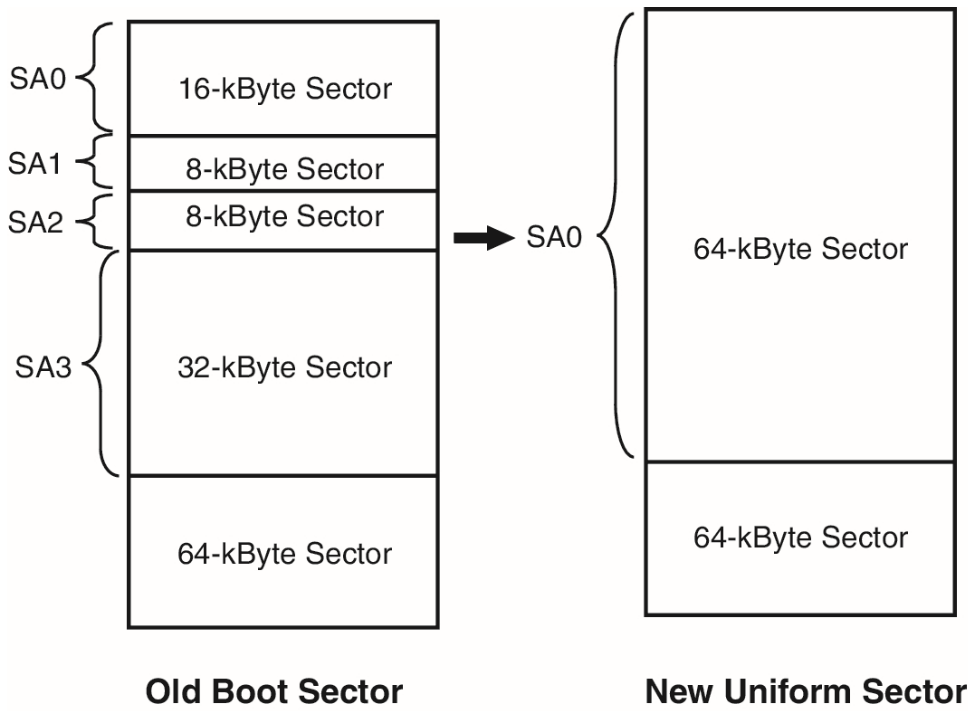
A small software change may be required. Instruct the device to erase the single 64-Kbyte sector. The command sequence to erase all of the boot sectors at once for the S29AL016D Model 02 should look something like this:

Address	555	2AA	555	555	2AA	SA0	SA1	SA2	SA3
Data	AA	55	80	AA	55	30	30	30	30

If the command sequence is used on a uniform device, the net effect is the same. The first 64 Kbytes of the Flash device, which SA0, SA1, SA2, & SA3 now point to, is erased once. The uniform Flash device ignores the additional sector erase commands since the commands essentially tell the Flash device to erase the first uniform sector four times. No code change is necessary. However the code will not be as clean as if the code had specifically been written for a uniform sectored device. In order to clean up the code, just remove the last three sector erase pulses. The modified sector erase command should look like the following sequence:

Address	555	2AA	555	555	2AA	SA0
Data	AA	55	80	AA	55	30

Where SA0 is now any address within the first 64-Kbyte uniform sector.



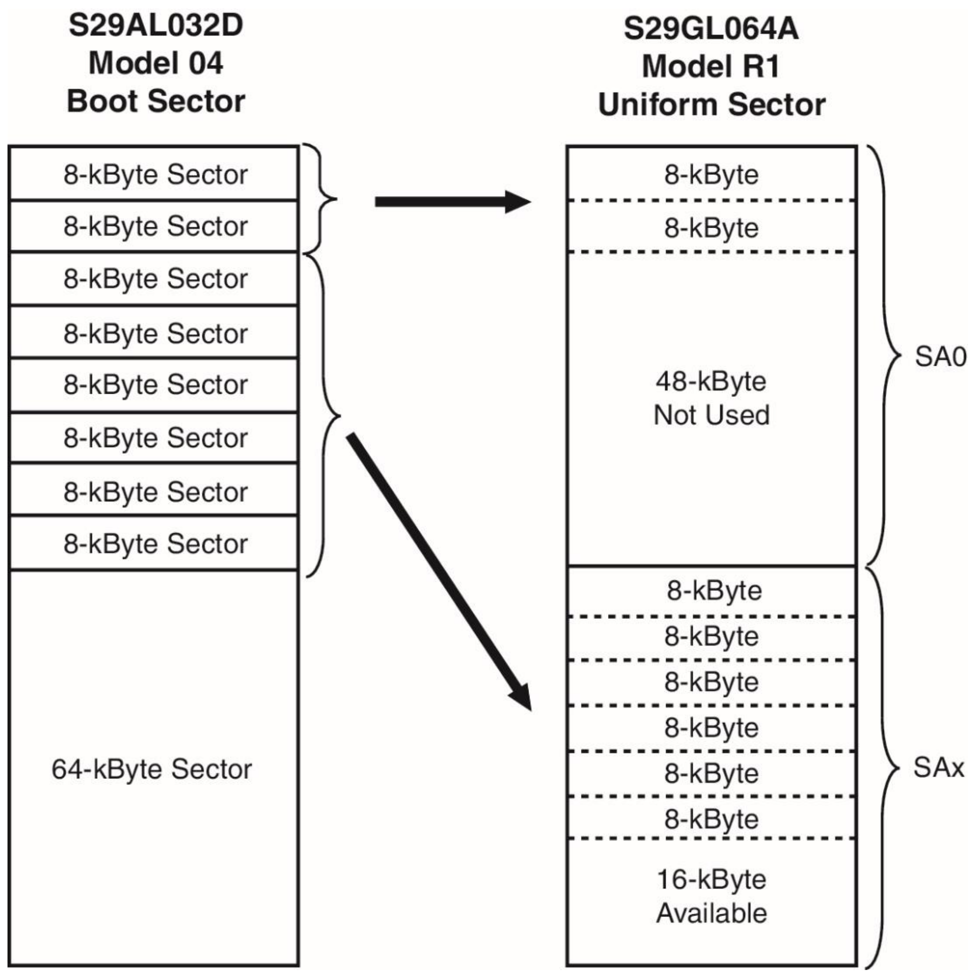
4.2 Case 2

The system requires individual boot sector erase capability.

Solution

Some address pointers for the smaller boot sectors need to be changed. To determine if the code currently in the boot sectors needs to be placed a separate 64-Kbyte sector, this question needs to be ask:

Does the code need to be modified by itself or can it be updated at the same time as other parts of the code? If it can be modified with other portions of the code, then portions of the code can be grouped together to potentially occupy all or part of a single 64-Kbyte sector. If it absolutely needs to be changed by itself, then it must be placed in its own 64-Kbyte sector. For each additional sector required the address pointers need to be changed. Let's look at an example:



In this case, let's assume a migration from a boot-sectored device to a uniform sectored device. The first two 8-Kbyte sectors are the boot code, which is never changed. The other 8-Kbyte boot sectors are used to store the operating code, which may need to be upgraded later. The boot code must be loaded into it's own 64-Kbyte sector and the addressing does not change. The remaining boot sectors need to be placed in one of the new 64-Kbyte sectors available in the larger memory device and the address pointers updated. Also please note that when migrating from boot to uniform it may be necessary to leave a small amount of memory space unused (48 Kbytes in the above figure) in order to retain the flexibility of the older boot sectored device.

5 Software Changes When Migrating From Uniform To Boot

5.1 Case 1

The system does not need to take advantage of the smaller boot sectors.

Solution

A small software change is needed to instruct the Flash device to erase all of the smaller sectors at once. This can be done by adding the sector addresses to the end of the normal sector erase command. Looking at an example going from uniform to bottom boot, the original sector erase command for a uniform sectored device would look like this:

Address	555	2AA	555	555	2AA	SA0
Data	AA	55	80	AA	55	30

Where SA0 is any address within the first 64-Kbyte sector. In order to convert to a bottom boot device, the sector erase command needs to be modified to look as follows:

Address	555	2AA	555	555	2AA	SA0	SA1	SA2	SA3
Data	AA	55	80	AA	55	30	30	30	30

Now SA0 points to any address within the first 16 Kbytes, SA1 points to an address within the next 8 Kbytes, SA2 points to the next 8 Kbytes, and finally SA3 points to the last 32 Kbytes. Together the four sector erases cover the first 64 Kbytes of the device. So this command will erase all of the small boot sectors at once, effectively making it one large uniform sector. All other sector erase commands will remain the same since the devices are identical past the first 64 Kbytes.

5.2 Case 2

The system takes advantage of the flexibility offered by the smaller boot sectors by erasing them independent of one another.

Solution

Software changes are necessary, but only to control erasing of the smaller sectors. Erasing of other sectors of the Flash device remain the same. For each small boot sector to be erased independently, the following command sequence needs to be included to identify the sector as well as specifying when the command will be called in the code.

Address	555	2AA	555	55	2AA	SA
Data	AA	55	80	AA	55	30

So with the uniform sector devices, only one erase command is needed for the first or last 64-Kbyte sector. With a boot sectored device you can have up to four different erase commands for a 16-Kbyte, 8-Kbyte, 8-Kbyte, and 32-Kbyte boot configuration or up to eight different sector erase commands for an eight 8-Kbyte boot configuration.

Please note that if a boot sectored device is accidentally used with code designed for a uniform sectored device, there could be potential problems. Some data that was thought to be erased, will still be present (since a sector erase issued to sector 0 will erase 64 Kbytes in the S29GL064A Model R1 device, but only erase a 8-Kbyte sector on the S29AL032D Model 04 device). Any attempt to program data from a “0” to a “1” in a sector which has not been erased will result in the Flash device going to an error state.

Revision history

Document version	Date of release	Description of changes
**	2006-11-15	Initial version
*A	2015-10-22	Updated in template
*B	2017-09-06	Updated logo and Copyright
*C	2021-03-30	Updated to Infineon template

Edition 2021-03-30

Published by

Infineon Technologies AG

81726 Munich, Germany

Do you have a question about this document?

Go to www.cypress.com/support

Document reference

001-99113 Rev. *C