# S29PL-N MirrorBit™ Flash Family

## 29PL256N, S29PL127N, S29PL129N,
## 256/128/128 Mb (16/8/8 M x 16-Bit) CMOS, 3.0 Volt-only Simultaneous Read/Write, Page-Mode Flash Memory

***Data Sheet*** *(Preliminary)*


**Notice to Readers:** This document states the current technical specifications regarding the Spansion product(s) described herein. Each product described herein may be designated as Advance Information, Preliminary, or Full Production. See *Notice On Data Sheet Designations* for definitions.

**Publication Number** S29PL-N_00 **Revision** A **Amendment** 5 **Issue Date** June 6, 2007

# Notice On Data Sheet Designations

Spansion Inc. issues data sheets with Advance Information or Preliminary designations to advise readers of product information or intended specifications throughout the product life cycle, including development, qualification, initial production, and full production. In all cases, however, readers are encouraged to verify that they have the latest information before finalizing their design. The following descriptions of Spansion data sheet designations are presented here to highlight their presence and definitions.

## Advance Information

The Advance Information designation indicates that Spansion Inc. is developing one or more specific products, but has not committed any design to production. Information presented in a document with this designation is likely to change, and in some cases, development on the product may discontinue. Spansion Inc. therefore places the following conditions upon Advance Information content:

> "This document contains information on one or more products under development at Spansion Inc. The information is intended to help you evaluate this product. Do not design in this product without contacting the factory. Spansion Inc. reserves the right to change or discontinue work on this proposed product without notice."

## Preliminary

The Preliminary designation indicates that the product development has progressed such that a commitment to production has taken place. This designation covers several aspects of the product life cycle, including product qualification, initial production, and the subsequent phases in the manufacturing process that occur before full production is achieved. Changes to the technical specifications presented in a Preliminary document should be expected while keeping these aspects of production under consideration. Spansion places the following conditions upon Preliminary content:

> "This document states the current technical specifications regarding the Spansion product(s) described herein. The Preliminary status of this document indicates that product qualification has been completed, and that initial production has begun. Due to the phases of the manufacturing process that require maintaining efficiency and quality, this document may be revised by subsequent versions or modifications due to changes in technical specifications."

## Combination

Some data sheets contain a combination of products with different designations (Advance Information, Preliminary, or Full Production). This type of document distinguishes these products and their designations wherever necessary, typically on the first page, the ordering information page, and pages with the DC Characteristics table and the AC Erase and Program table (in the table notes). The disclaimer on the first page refers the reader to the notice on this page.

## Full Production (No Designation on Document)

When a product has been in production for a period of time such that no changes or only nominal changes are expected, the Preliminary designation is removed from the data sheet. Nominal changes may include those affecting the number of ordering part numbers available, such as the addition or deletion of a speed option, temperature range, package type, or $V_{IO}$ range. Changes may also include those needed to clarify a description or to correct a typographical error or incorrect specification. Spansion Inc. applies the following conditions to documents in this category:

> "This document states the current technical specifications regarding the Spansion product(s) described herein. Spansion Inc. deems the products to have been in sufficient production volume such that subsequent versions of this document are not expected to change. However, typographical or specification corrections, or modifications to the valid combinations offered may occur."

Questions regarding these document designations may be directed to your local sales office.

# S29PL-N MirrorBit™ Flash Family

## 29PL256N, S29PL127N, S29PL129N,
## 256/128/128 Mb (16/8/8 M x 16-Bit) CMOS, 3.0 Volt-only Simultaneous Read/Write, Page-Mode Flash Memory

***Data Sheet*** *(Preliminary)*

## General Description

The Spansion Family Name is the latest generation 3.0-Volt page mode read family fabricated using the 110 nm Mirrorbit™ Flash process technology. These 8-word page-mode Flash devices are capable of performing simultaneous read and write operations with zero latency on two separate banks. These devices offer fast page access times of 25 to 30 ns, with corresponding random access times of 65 ns, 70 ns, and 80 ns respectively, allowing high speed microprocessors to operate without wait states. The S29PL129N device offers the additional feature of dual chip enable inputs (CE1# and CE2#) that allow each half of the memory space to be controlled separately.

## Distinctive Characteristics

### Architectural Advantages

- **32-Word Write Buffer**
- **Dual Chip Enable Inputs (only for S29PL129N)**
  - Two CE# inputs control selection of each half of the memory space
- **Single Power Supply Operation**
  - Full Voltage range of 2.7 – 3.6 V read, erase, and program operations for battery-powered applications
  - Voltage range of 2.7 – 3.1 V valid for PL-N MCP products
- **Simultaneous Read/Write Operation**
  - Data can be continuously read from one bank while executing erase/program functions in another bank
  - Zero latency switching from write to read operations
- **4-Bank Sector Architecture with Top and Bottom Boot Blocks**
- **256-Word Secured Silicon Sector Region**
  - Up to 128 factory-locked words
  - Up to 128 customer-lockable words
- **Manufactured on 0.11 µm Process Technology**
- **Data Retention of 20 years Typical**
- **Cycling Endurance of 100,000 Cycles per Sector Typical**

### Hardware Features

- **WP#/ACC (Write Protect/Acceleration) Input**
  - At $V_{IL}$, hardware level protection for the first and last two 32 Kword sectors.
  - At $V_{IH}$, allows the use of DYB/PPB sector protection
  - At $V_{HH}$, provides accelerated programming in a factory setting
- **Dual Boot and No Boot Options**
- **Low $V_{CC}$ Write Inhibit**

### Security Features

- **Persistent Sector Protection**
  - A command sector protection method to lock combinations of individual sectors to prevent program or erase operations within that sector
  - Sectors can be locked and unlocked in-system at $V_{CC}$ level
- **Password Sector Protection**
  - A sophisticated sector protection method locks combinations of individual sectors to prevent program or erase operations within that sector using a user defined 64-bit password

**Publication Number** S29PL-N_00 **Revision** A **Amendment** 5 **Issue Date** June 6, 2007

# Performance Characteristics

| Read Access Times (@ 30 pF, Industrial Temp.) | | |
|---|---|---|
| Random Access Time, ns ($t_{ACC}$) | 65 | 70 |
| Page Access Time, ns ($t_{PACC}$) | 25 | 30 |
| Max CE# Access Time, ns ($t_{CE}$) | 65 | 70 |
| Max OE# Access Time, ns ($t_{OE}$) | 25 | 30 |

| Current Consumption (typical values) | |
|---|---|
| 8-Word Page Read | 6 mA |
| Simultaneous Read/Write | 65 mA |
| Program/Erase | 25 mA |
| Standby | 20 µA |

| Typical Program & Erase Times (typical values) (See Note) | |
|---|---|
| Typical Word | 40 µs |
| Typical Effective Word (32 words in buffer) | 9.4 µs |
| Accelerated Write Buffer Program | 6 µs |
| Typical Sector Erase Time (32-Kword Sector) | 300 ms |
| Typical Sector Erase Time (128-Kword Sector) | 1.6 s |

***Note***
*Typical program and erase times assume the following conditions: 25°C, 3.0 V $V_{CC}$, 10,000 cycles; checkerboard data pattern.*

| Package Options | | |
|---|---|---|
| **S29PL-N** | **VBH064 8.0 x 11.6 mm, 64-ball** | **VBH084 8.0 x 11.6 mm, 84-ball** |
| 256 | | ● |
| 129 | ● | |
| 127 | ● | |

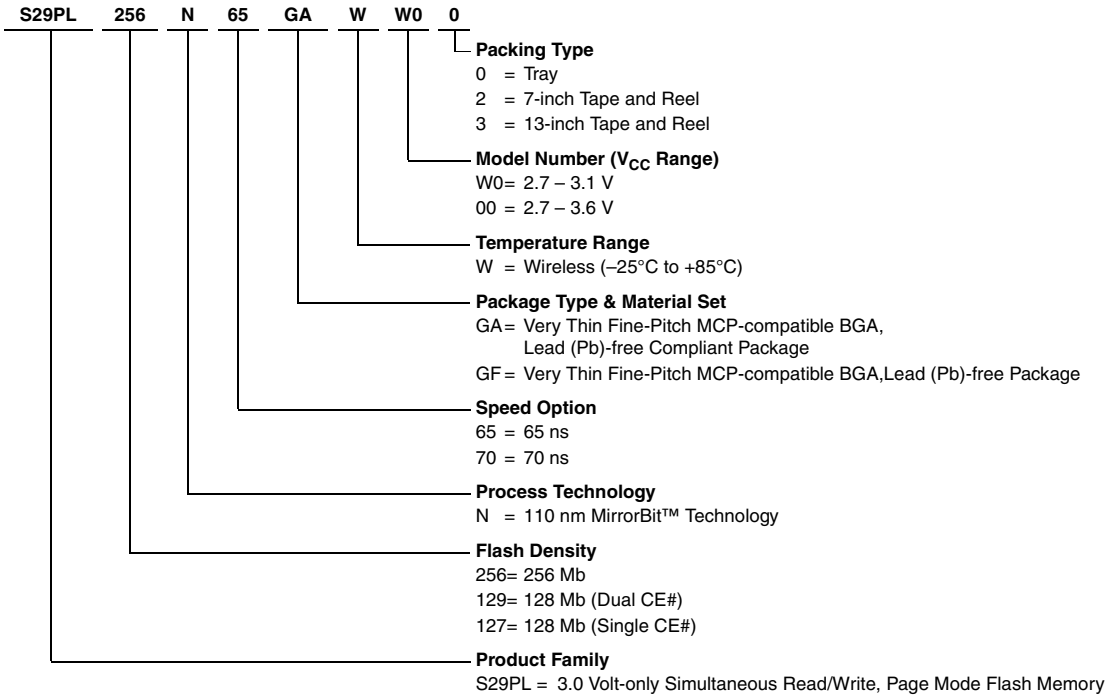
# Table of Contents

# Figures

# Tables

# 1. Ordering Information

The ordering part number is formed by a valid combination of the following:




| Valid Combinations | | | | | $V_{IO}$ Range | Package Type (Note 2) |
|---|---|---|---|---|---|---|
| Base Ordering Part Number | Speed Option | Package Type, Material, & Temperature Range | Model Number | Packing Type | | |
| S29PL256N | 65, 70 | GAW, GFW | W0 | 0, 2, 3 (Note 1) | 2.7 – 3.1 V | VBH**084** 8.0 x 11.6 mm **84**-ball MCP-Compatible (FBGA) |
| S29PL127N<br>S29PL129N | | | | | | VBH**064** 8.0 x 11.6 mm **64**-ball MCP-Compatible (FBGA) |

***Notes***

1. *Type 0 is standard. Specify other options as required.*
2. *BGA package marking omits leading S29 and packing type designator from ordering part number.*

## Valid Combinations

Valid Combinations list configurations planned to be supported in volume for this device. Consult your local sales office to confirm availability of specific valid combinations and to check on newly released combinations.

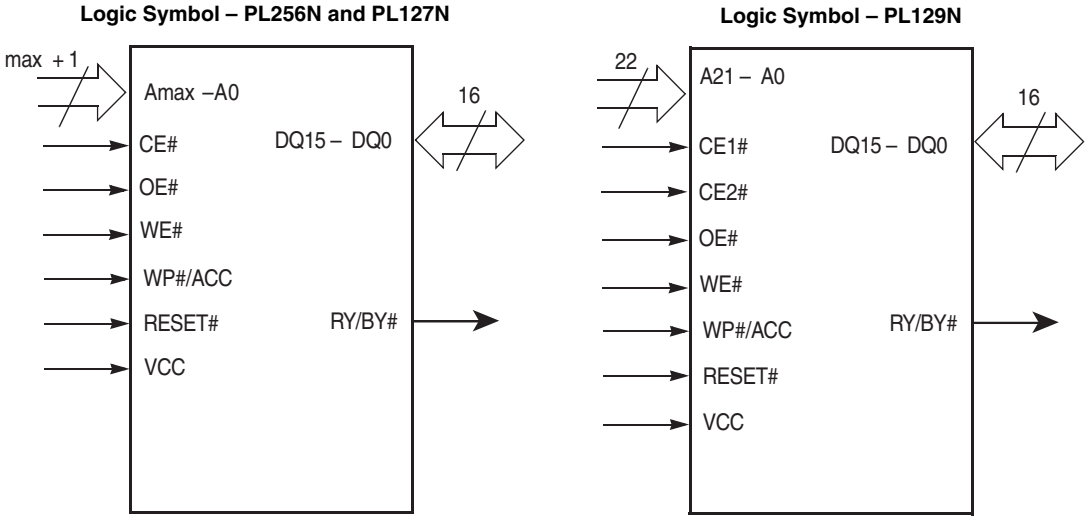
# 2. Input/Output Descriptions and Logic Symbols

Table 2.1 identifies the input and output package connections provided on the device.

**Table 2.1** Input/Output Descriptions

| Symbol | Type | Description |
|---|---|---|
| $A_{max}$ – A0 | Input | Address bus |
| DQ15 – DQ0 | I/O | 16-bit data inputs/outputs/float |
| CE# | Input | Chip Enable input |
| OE# | Input | Output Enable input |
| WE# | Input | Write Enable |
| $V_{SS}$ | Supply | Device ground |
| NC | Not connected | Pin Not Connected Internally |
| RY/BY# | Output | Ready/Busy output and open drain.<br>When RY/BY#= $V_{IH}$, the device is ready to accept read operations and commands.<br>When RY/BY#= $V_{OL}$, the device is either executing an embedded algorithm or the device is executing a hardware reset operation. |
| $V_{CC}$ | Supply | Device Power Supply |
| RESET# | Input | Hardware reset pin |
| CE1#, CE2# | Input | Chip Enable inputs for S29PL129 device |

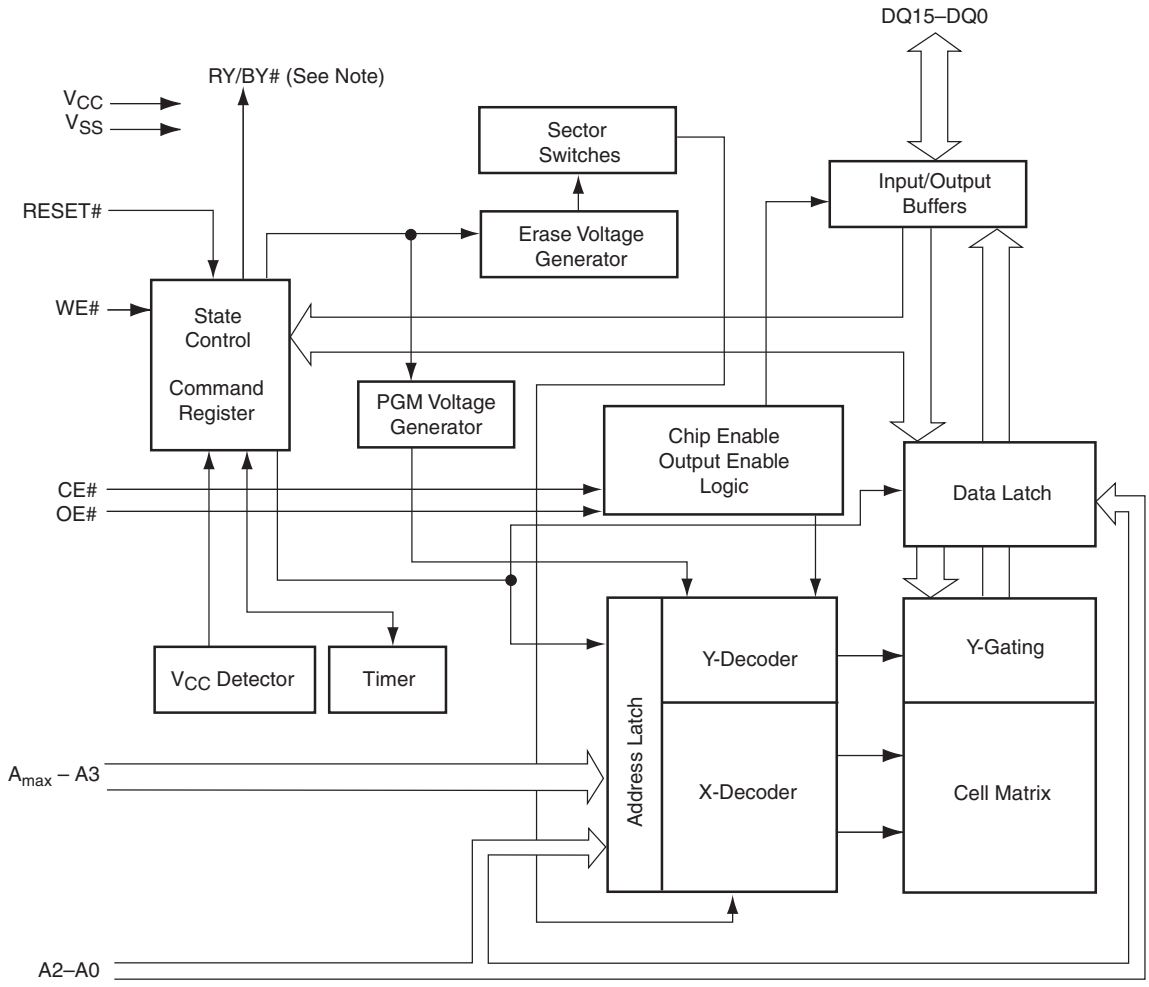
**Figure 2.1** Logic Symbols – PL256N, PL129N, and PL127N




***Note***

*1. Amax = 23 for the PL256N and 22 for the PL127N.*

# 3. Block Diagram




***Notes***

1. *RY/BY# is an open drain output.*
2. *$A_{max}$ = A23 (PL256N), A22 (PL127N), A21 (PL129N).*
3. *PL129N has two CE# pins CE1# and CE2#.*

# 4. Connection Diagrams/Physical Dimensions

This section contains the I/O designations and package specifications for the OPN.

## 4.1 Special Handling Instructions for FBGA Package

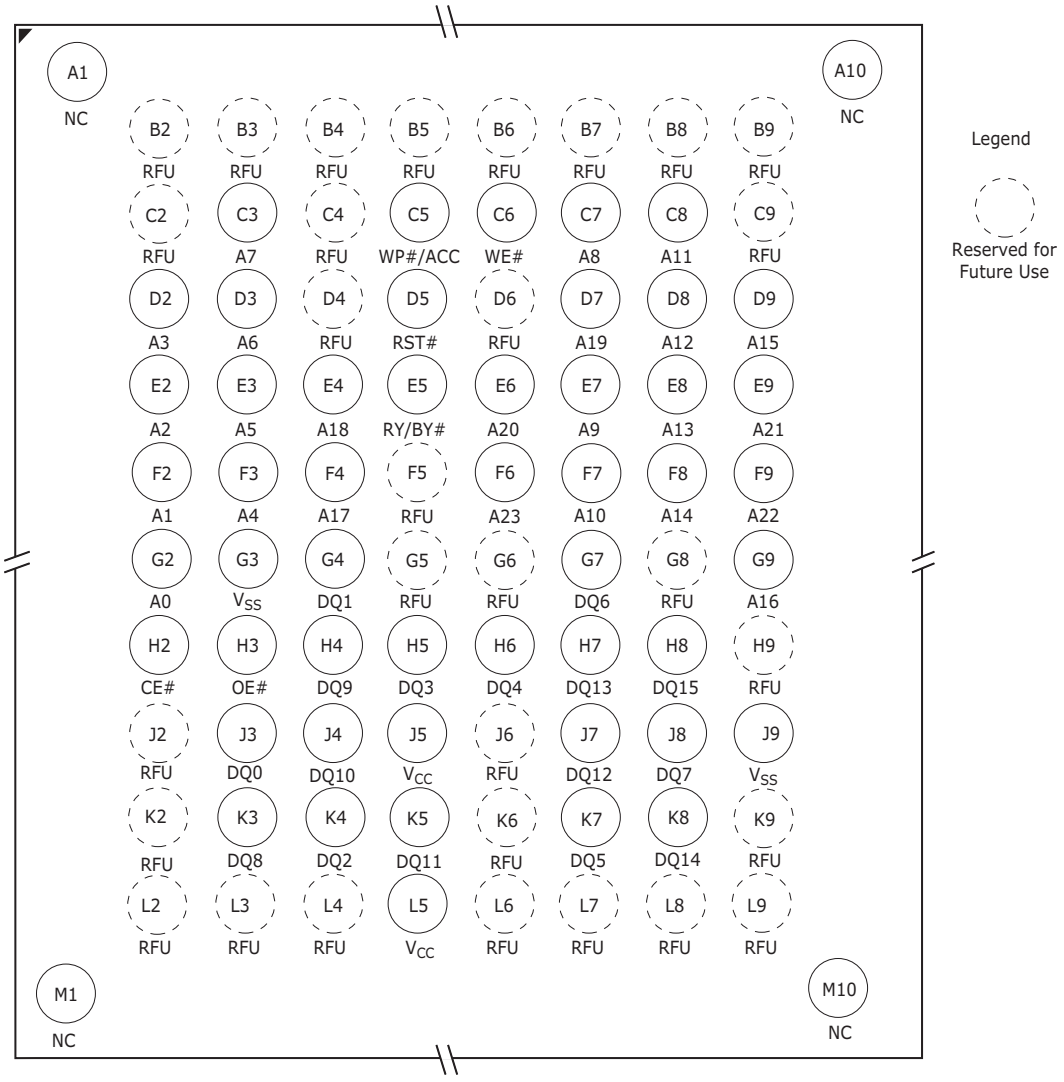
Special handling is required for Flash Memory products in FBGA packages.

Flash memory devices in FBGA packages may be damaged if exposed to ultrasonic cleaning methods. The package and/or data integrity may be compromised if the package body is exposed to temperatures above 150°C for prolonged periods of time.

## 4.2 VBH084, 8.0 x 11.6 mm

### 4.2.1 Connection Diagram – S29PL256N MCP Compatible Package

**Figure 4.1** Connection Diagram – 84-ball Fine-Pitch Ball Grid Array (S29PL256N)

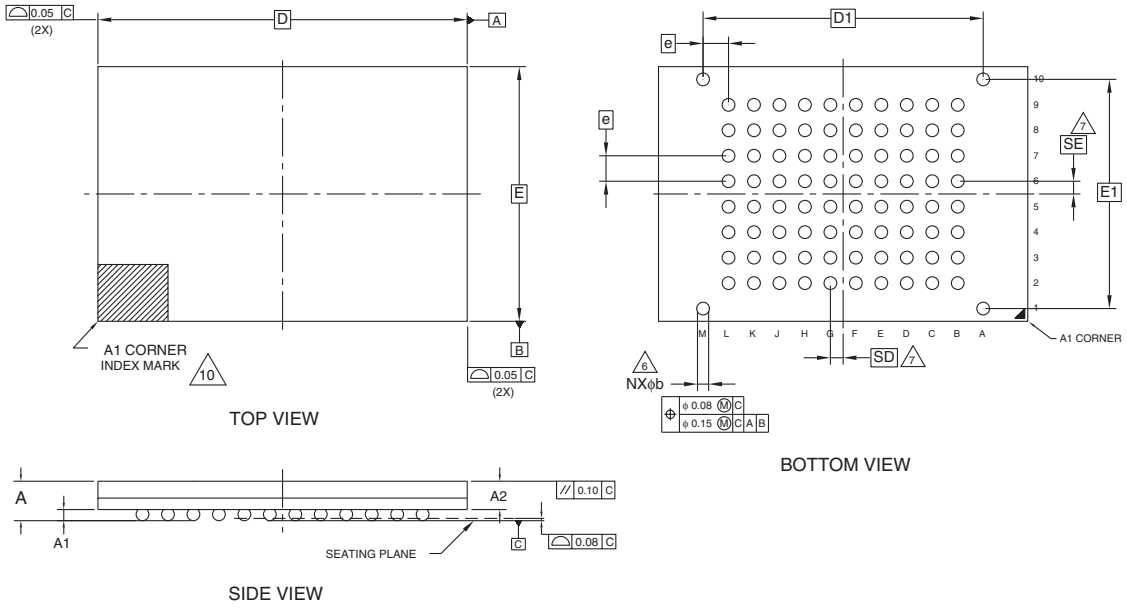



***Notes***

1. *Top view—balls facing down.*
2. *Recommended for wireless applications*

## 4.2.2 Physical Dimensions – VBH084, 8.0 x 11.6 mm

**Figure 4.2** Physical Dimensions – 84-ball Fine-Pitch Ball Grid Array (S29PL256N)




| PACKAGE | VBH 084 | | | |
|---|---|---|---|---|
| JEDEC | N/A | | | |
| | 11.60 mm x 8.00 mm NOM PACKAGE | | | |
| SYMBOL | MIN | NOM | MAX | NOTE |
| A | --- | --- | 1.00 | OVERALL THICKNESS |
| A1 | 0.18 | --- | --- | BALL HEIGHT |
| A2 | 0.62 | --- | 0.76 | BODY THICKNESS |
| D | 11.60 BSC. | | | BODY SIZE |
| E | 8.00 BSC. | | | BODY SIZE |
| D1 | 8.80 BSC. | | | BALL FOOTPRINT |
| E1 | 7.20 BSC. | | | BALL FOOTPRINT |
| MD | 12 | | | ROW MATRIX SIZE D DIRECTION |
| ME | 10 | | | ROW MATRIX SIZE E DIRECTION |
| N | 84 | | | TOTAL BALL COUNT |
| ϕb | 0.33 | --- | 0.43 | BALL DIAMETER |
| e | 0.80 BSC. | | | BALL PITCH |
| SD / SE | 0.40 BSC. | | | SOLDER BALL PLACEMENT |
| | (A2-A9, B10-L10, M2-M9, B1-L1) | | | DEPOPULATED SOLDER BALLS |

NOTES:

1. DIMENSIONING AND TOLERANCING PER ASME Y14.5M-1994.
2. ALL DIMENSIONS ARE IN MILLIMETERS.
3. BALL POSITION DESIGNATION PER JESD 95-1, SPP-010 (EXCEPT AS NOTED).
4. e REPRESENTS THE SOLDER BALL GRID PITCH.
5. SYMBOL "MD" IS THE BALL ROW MATRIX SIZE IN THE "D" DIRECTION.

   SYMBOL "ME" IS THE BALL COLUMN MATRIX SIZE IN THE "E" DIRECTION.

   N IS THE TOTAL NUMBER OF SOLDER BALLS.
6. DIMENSION "b" IS MEASURED AT THE MAXIMUM BALL DIAMETER IN A PLANE PARALLEL TO DATUM C.
7. SD AND SE ARE MEASURED WITH RESPECT TO DATUMS A AND B AND DEFINE THE POSITION OF THE CENTER SOLDER BALL IN THE OUTER ROW.

   WHEN THERE IS AN ODD NUMBER OF SOLDER BALLS IN THE OUTER ROW PARALLEL TO THE D OR E DIMENSION, RESPECTIVELY, SD OR SE = 0.000.

   WHEN THERE IS AN EVEN NUMBER OF SOLDER BALLS IN THE OUTER ROW, SD OR SE = e/2
8. NOT USED.
9. "+" INDICATES THE THEORETICAL CENTER OF DEPOPULATED BALLS.
10. A1 CORNER TO BE IDENTIFIED BY CHAMFER, LASER OR INK MARK, METALLIZED MARK INDENTATION OR OTHER MEANS.

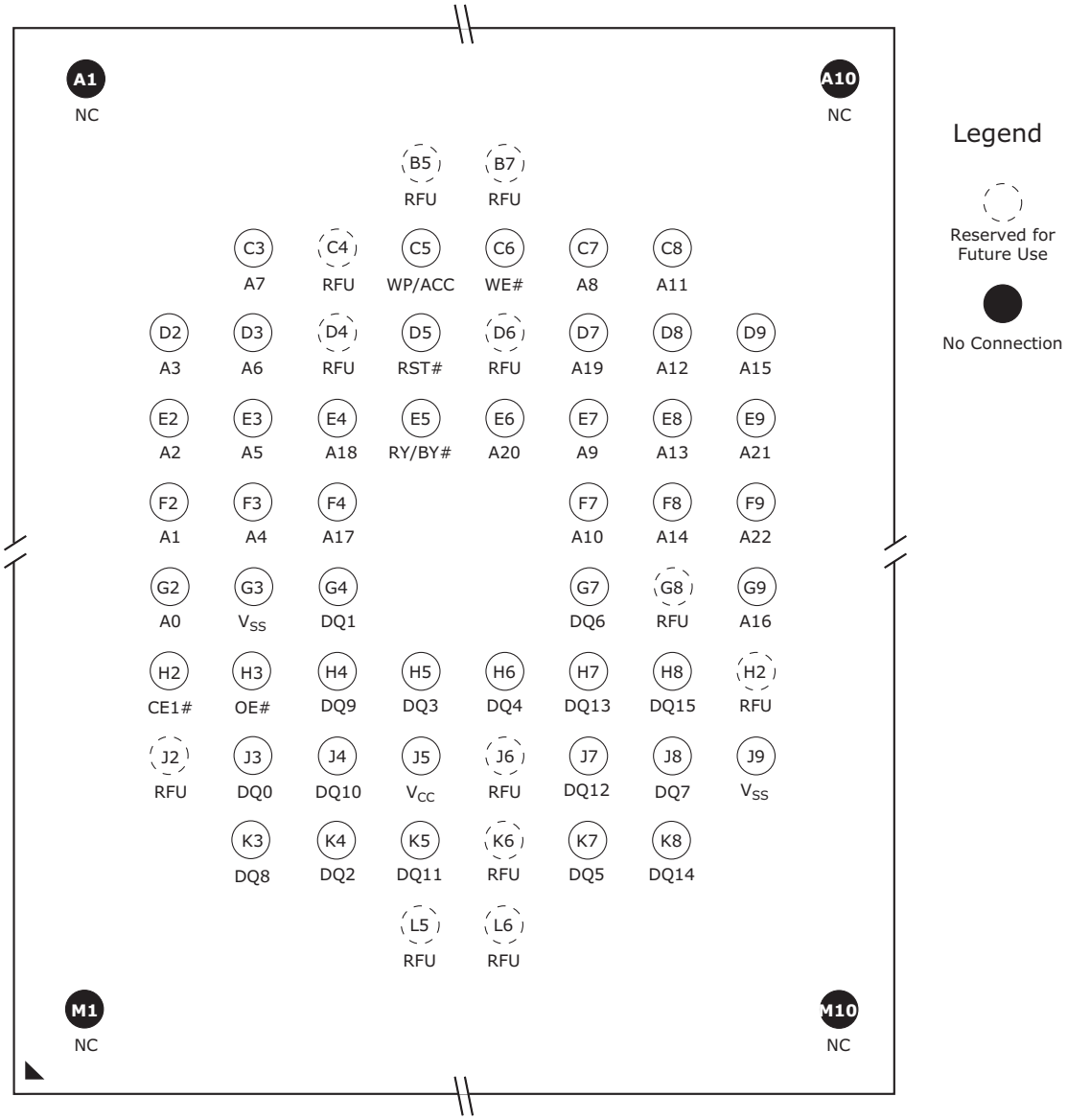
3339 \ 16-038.25b

***Note***
*Recommended for wireless applications*

## 4.3 VBH064, 8 x 11.6 mm

### 4.3.1 Connection Diagram – S29PL127N MCP Compatible Package

**Figure 4.3** Connection Diagram – 64-Ball Fine-Pitch Ball Grid Array (S29PL127N)

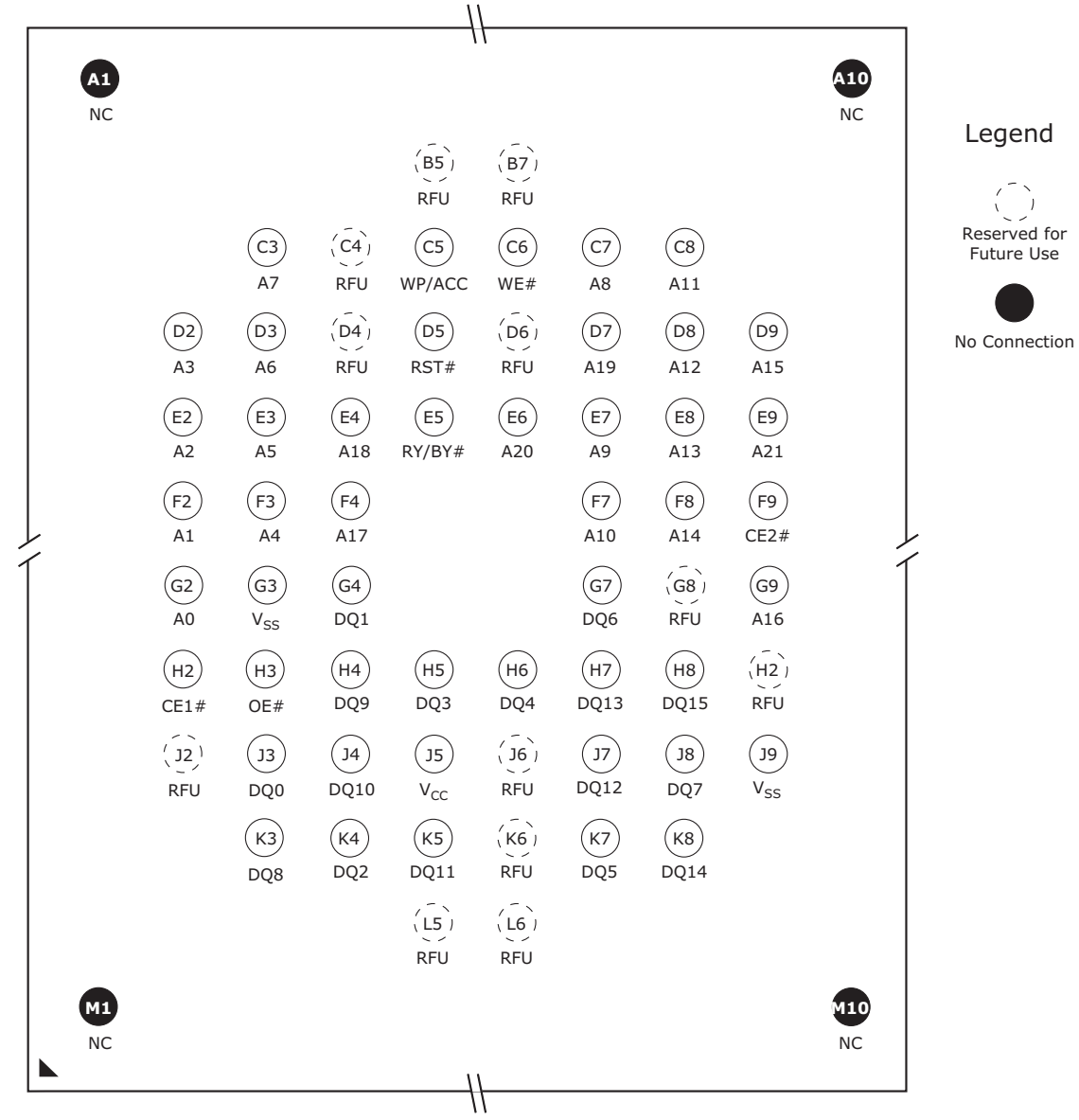



***Notes***

1. *Top view—balls facing down.*
2. *Recommended for wireless applications*

### 4.3.2 Connection Diagram – S29PL129N MCP Compatible Package

**Figure 4.4** Connection Diagram – 64-Ball Fine-Pitch Ball Grid Array (S29PL129N)

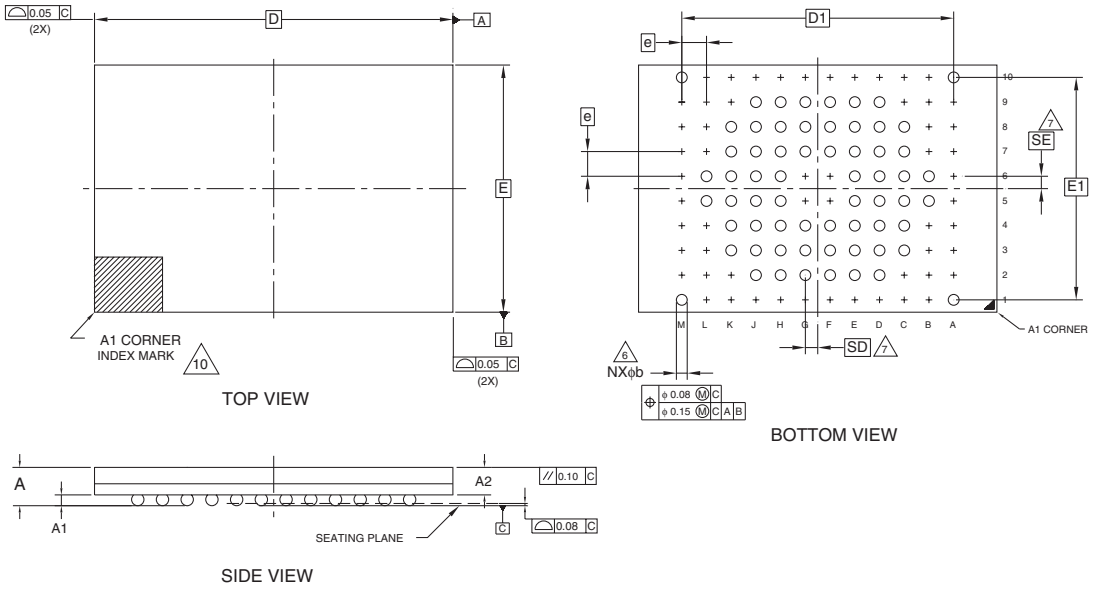



***Notes***

1. *Top view—balls facing down.*
2. *Recommended for wireless applications*

## 4.3.3 Physical Dimensions – VBH064, 8 x 11.6 mm – S29PL-N

**Figure 4.5** Physical Dimensions – 64-Ball Fine-Pitch Ball Grid Array (S29PL-N)




| PACKAGE | VBH 064 | | | |
|---|---|---|---|---|
| JEDEC | N/A | | | |
| | 11.60 mm x 8.00 mm NOM PACKAGE | | | |
| SYMBOL | MIN | NOM | MAX | NOTE |
| A | --- | --- | 1.00 | OVERALL THICKNESS |
| A1 | 0.18 | --- | --- | BALL HEIGHT |
| A2 | 0.62 | --- | 0.76 | BODY THICKNESS |
| D | 11.60 BSC. | | | BODY SIZE |
| E | 8.00 BSC. | | | BODY SIZE |
| D1 | 8.80 BSC. | | | BALL FOOTPRINT |
| E1 | 7.20 BSC. | | | BALL FOOTPRINT |
| MD | 12 | | | ROW MATRIX SIZE D DIRECTION |
| ME | 10 | | | ROW MATRIX SIZE E DIRECTION |
| N | 64 | | | TOTAL BALL COUNT |
| ϕb | 0.33 | --- | 0.43 | BALL DIAMETER |
| e | 0.80 BSC. | | | BALL PITCH |
| SD / SE | 0.40 BSC. | | | SOLDER BALL PLACEMENT |
| | (A2-9,B1-4,B7-10,C1-K1, M2-9,C10-K10,L1-4,L7-10, G5-6,F5-6) | | | DEPOPULATED SOLDER BALLS |

NOTES:

1. DIMENSIONING AND TOLERANCING PER ASME Y14.5M-1994.
2. ALL DIMENSIONS ARE IN MILLIMETERS.
3. BALL POSITION DESIGNATION PER JESD 95-1, SPP-010 (EXCEPT AS NOTED).
4. e REPRESENTS THE SOLDER BALL GRID PITCH.
5. SYMBOL "MD" IS THE BALL ROW MATRIX SIZE IN THE "D" DIRECTION.

   SYMBOL "ME" IS THE BALL COLUMN MATRIX SIZE IN THE "E" DIRECTION.

   N IS THE TOTAL NUMBER OF SOLDER BALLS.
6. DIMENSION "b" IS MEASURED AT THE MAXIMUM BALL DIAMETER IN A PLANE PARALLEL TO DATUM C.
7. SD AND SE ARE MEASURED WITH RESPECT TO DATUMS A AND B AND DEFINE THE POSITION OF THE CENTER SOLDER BALL IN THE OUTER ROW.

   WHEN THERE IS AN ODD NUMBER OF SOLDER BALLS IN THE OUTER ROW PARALLEL TO THE D OR E DIMENSION, RESPECTIVELY, SD OR SE = 0.000.

   WHEN THERE IS AN EVEN NUMBER OF SOLDER BALLS IN THE OUTER ROW, SD OR SE = e/2
8. NOT USED.
9. "+" INDICATES THE THEORETICAL CENTER OF DEPOPULATED BALLS.
10. A1 CORNER TO BE IDENTIFIED BY CHAMFER, LASER OR INK MARK, METALLIZED MARK INDENTATION OR OTHER MEANS.

3330 \ 16-038.25b

***Note***
*Recommended for wireless applications*

# 5. Additional Resources

Visit www.spansion.com to obtain the following related documents:

## 5.1 Application Notes

- Using the Operation Status Bits in Spansion Devices
- Simultaneous Read/Write vs. Erase Suspend/Resume
- MirrorBit™ Flash Memory Write Buffer Programming and Page Buffer Read
- Design-In Scalable Wireless Solutions with Spansion Products
- Common Flash Interface Version 1.4 Vendor Specific Extensions

## 5.2 Specification Bulletins

Contact your local sales office for details.

### Drivers and Software Support

- Spansion Low-Level Drivers
- Enhanced Flash Drivers
- Flash File System

### CAD Modeling Support

- VHDL and Verilog
- IBIS
- ORCAD

## 5.3 Technical Support

Contact your local sales office or contact Spansion Inc. directly for additional technical support:

### Email

US and Canada: HW.support@amd.com
Asia Pacific: asia.support@amd.com
Europe, Middle East, and Africa
Japan: http://edevice.fujitsu.com/jp/support/tech/#b7

### Frequently Asked Questions (FAQ)

http://ask.amd.com/
http://edevice.fujitsu.com/jp/support/tech/#b7

### Phone

US: (408) 749-5703
Japan (03) 5322-3324

### Spansion Inc. Locations

915 DeGuigne Drive, P.O. Box 3453
Sunnyvale, CA 94088-3453, USA
Telephone: 408-962-2500 or
1-866-SPANSION

Spansion Japan Limited
4-33-4 Nishi Shinjuku, Shinjuku-ku
Tokyo, 160-0023
Telephone: +81-3-5302-2200
Facsimile: +81-3-5302-2674

http://www.spansion.com

# 6. Product Overview

The S29PL-N family consists of 256 and 128 Mb, 3.0 volts-only, simultaneous read/write page-mode read Flash devices that are optimized for wireless designs of today that demand large storage array and rich functionality, while requiring low power consumption. These products also offer 32-word buffer for programming with program and erase suspend/resume functionality. Additional features include:

- Advanced Sector Protection methods for protecting an individual or group of sectors as required,
- 256-word of secured silicon area for storing customer and factory secured information
- Simultaneous Read/Write operation

## 6.1 Memory Map

The S29PL-N devices consist of 4 banks organized as shown in Table 6.1, 6.2, and 6.3.

**Table 6.1** PL256N Sector and Memory Address Map

| Bank | Bank Size | Sector Count | Sector Size (KB) | Sector/ Sector Range | Address Range | Notes |
|---|---|---|---|---|---|---|
| A | 4 MB | 4 | 64 | SA00 | 000000h-007FFFh | Sector Starting Address – Sector Ending Address |
| | | | 64 | SA01 | 008000h-00FFFFh | |
| | | | 64 | SA02 | 010000h-017FFFh | |
| | | | 64 | SA03 | 018000h-01FFFFh | |
| | | 15 | 256 | SA04 | 020000h-03FFFFh | Sector Starting Address - Sector Ending Address (see note) |
| | | | ⋮ | ⋮ | ⋮ | |
| | | | 256 | SA018 | 1E0000h-1FFFFFh | |
| B | 12 MB | 48 | 256 | SA19 | 200000h-21FFFFh | First Sector, Sector Starting Address - Last Sector, Sector Ending Address (see note) |
| | | | ⋮ | ⋮ | ⋮ | |
| | | | 256 | SA66 | 7E0000h-7FFFFFh | |
| C | 12 MB | 48 | 256 | SA67 | 800000h-81FFFFh | First Sector, Sector Starting Address - Last Sector, Sector Ending Address (see note) |
| | | | ⋮ | ⋮ | ⋮ | |
| | | | 256 | SA114 | DE0000h-DFFFFFh | |
| D | 4 MB | 15 | 256 | SA115 | E00000h-E1FFFFh | Sector Starting Address - Sector Ending Address (see note) |
| | | | ⋮ | ⋮ | ⋮ | |
| | | | 256 | SA129 | FC0000h-FDFFFFh | |
| | | 4 | 64 | SA130 | FE0000h-FE7FFFh | Sector Starting Address - Sector Ending Address |
| | | | 64 | SA131 | FE8000h-FEFFFFh | |
| | | | 64 | SA132 | FF0000h-FF7FFFh | |
| | | | 64 | SA133 | FF8000h-FFFFFFh | |

***Note***
*Ellipses indicate that other addresses in sector range follow the same pattern.*

**Table 6.2** PL127N Sector and Memory Address Map

| Bank | Bank Size | Sector Count | Sector Size (KB) | Sector/ Sector Range | Address Range | Notes |
|---|---|---|---|---|---|---|
| A | 2 MB | 4 | 64 | SA00 | 000000h-007FFFh | Sector Starting Address - Sector Ending Address |
| | | | 64 | SA01 | 008000h-00FFFFh | |
| | | | 64 | SA02 | 010000h-017FFFh | |
| | | | 64 | SA03 | 018000h-01FFFFh | |
| | | 7 | 256 | SA04 | 020000h-03FFFFh | Sector Starting Address – Sector Ending Address (see note) |
| | | | ⋮ | ⋮ | ⋮ | |
| | | | 256 | SA10 | 0E0000h-0FFFFFh | |
| B | 6 MB | 24 | 256 | SA11 | 100000h-11FFFFh | First Sector, Sector Starting Address - Last Sector, Sector Ending Address (see note) |
| | | | ⋮ | ⋮ | ⋮ | |
| | | | 256 | SA34 | 3E0000h-3FFFFFh | |
| C | 6 MB | 24 | 256 | SA35 | 400000h-41FFFFh | First Sector, Sector Starting Address - Last Sector, Sector Ending Address (see note) |
| | | | ⋮ | ⋮ | ⋮ | |
| | | | 256 | SA58 | 6E0000h-6FFFFFh | |
| D | 2 MB | 7 | 256 | SA59 | 700000h-71FFFFh | Sector Starting Address - Sector Ending Address (see note) |
| | | | ⋮ | ⋮ | ⋮ | |
| | | | 256 | SA65 | 7C0000h-7DFFFFh | |
| | | 4 | 64 | SA66 | 7E0000h-7E7FFFh | Sector Starting Address - Sector Ending Address |
| | | | 64 | SA67 | 7E80000h-7EFFFFh | |
| | | | 64 | SA68 | 7F0000h-7F7FFFh | |
| | | | 64 | SA69 | 7F8000h-7FFFFFh | |

***Note***
*Ellipses indicate that other addresses in sector range follow the same pattern.*

**Table 6.3** PL129N Sector and Memory Address Map

| Bank | Bank Size | Sector Count | Sector Size (KB) | CE1# | CE2# | Sector/ Sector Range | Address Range | Notes |
|---|---|---|---|---|---|---|---|---|
| 1A | 2 MB | 4 | 64 | $V_{IL}$ | $V_{IH}$ | SA00 | 000000h-007FFFh | Sector Starting Address - Sector Ending Address |
| | | | 64 | | | SA01 | 008000h-00FFFFh | |
| | | | 64 | | | SA02 | 010000h-017FFFh | |
| | | | 64 | | | SA03 | 018000h-01FFFFh | |
| | | 7 | 256 | | | SA04 | 020000h-03FFFFh | Sector Starting Address – Sector Ending Address (see note) |
| | | | ⋮ | | | ⋮ | ⋮ | |
| | | | 256 | | | SA10 | 0E0000h-0FFFFFh | |
| 1B | 6 MB | 24 | 256 | | | SA11 | 100000h-11FFFFh | First Sector, Sector Starting Address - Last Sector, Sector Ending Address (see note) |
| | | | ⋮ | | | ⋮ | ⋮ | |
| | | | 256 | | | SA34 | 3E0000h-3FFFFFh | |
| 2A | 6 MB | 24 | 256 | $V_{IH}$ | $V_{IL}$ | SA35 | 000000h-01FFFFh | First Sector, Sector Starting Address - Last Sector, Sector Ending Address (see note) |
| | | | ⋮ | | | ⋮ | ⋮ | |
| | | | 256 | | | SA58 | 2E0000h - 2FFFFFh | |
| 2B | 2 MB | 7 | 256 | | | SA59 | 300000h-31FFFFh | Sector Starting Address - Sector Ending Address (see note) |
| | | | ⋮ | | | ⋮ | ⋮ | |
| | | | 256 | | | SA65 | 3C0000h-3DFFFFh | |
| | | 4 | 64 | | | SA66 | 3E0000h-3E7FFFH | Sector Starting Address - Sector Ending Address |
| | | | 64 | | | SA67 | 3E8000h-3EFFFFh | |
| | | | 64 | | | SA68 | 3F0000h-3F7FFFh | |
| | | | 64 | | | SA69 | 3F8000h-3FFFFFh | |

# 7. Device Operations

This section describes the read, program, erase, simultaneous read/write operations, and reset features of the Flash devices.

Operations are initiated by writing specific commands or a sequence with specific address and data patterns into the command registers (see Table 12.1 on page 66 and Table 12.2 on page 68). The command register itself does not occupy any addressable memory location. Instead, the command register is composed of latches that store the commands, along with the address and data information needed to execute the command. The contents of the register serve as input to the internal state machine and the state machine outputs dictate the function of the device. Writing incorrect address and data values or writing them in an improper sequence can place the device in an unknown state, in which case the system must write the reset command to return the device to the reading array data mode.

## 7.1 Device Operation Table

The device must be setup appropriately for each operation. Table 7.1 describes the required state of each control pin for any particular operation.

**Table 7.1** Device Operation

| Operation | CE# | OE# | WE# | RESET# | WP#/ACC | Addresses ($A_{max}$ – A0) | DQ15 – DQ0 |
|---|---|---|---|---|---|---|---|
| Read | L | L | H | H | X | $A_{IN}$ | $D_{OUT}$ |
| Write | L | H | L | H | X (See Note) | $A_{IN}$ | $D_{IN}$ |
| Standby | H | X | X | H | X | $A_{IN}$ | High-Z |
| Output Disable | L | H | H | H | X | $A_{IN}$ | High-Z |
| Reset | X | X | X | L | X | $A_{IN}$ | High-Z |

***Legend***

*L = Logic Low = $V_{IL}$* — *H = Logic High = $V_{IH}$*

*$V_{HH}$ = 8.5 – 9.5 V* — *X = Don't Care*

*SA = Sector Address* — *$A_{IN}$ = Address In*

*$D_{IN}$ = Data In* — *$D_{OUT}$ = Data Out*

***Note***

*WP#/ACC must be high when writing to upper two and lower two sectors (PL256N: 0, 1,132, and 133; PL127/129N: 0, 1, 68, and 69)*

### 7.1.1 Dual Chip Enable Device Description and Operation (PL129N Only)

The dual CE# product (PL129N) offers a reduced number of address pins to accommodate processors with a limited addressable range. This product operates as two separate devices in a single package and requires the processor to address half of the memory space with one chip enable and the remaining memory space with a second chip enable. For more details on the addressing features of the Dual CE# device refer to Table 6.3 on page 19 for the PL129N Sector and Memory Address Map.

Dual chip enable products must be setup appropriately for each operation. To place the device into the active state either CE1# or CE2# must be set to $V_{IL}$. To place the device in standby mode, both CE1# and CE2# must be set to $V_{IH}$. Table 7.2 describes the required state of each control pin for any particular operation.

**Table 7.2** Dual Chip Enable Device Operation

| Operation | CE1# | CE2# | OE# | WE# | RESET# | WP#/ACC | Addresses (A21 – A0) | DQ15 – DQ0 |
|---|---|---|---|---|---|---|---|---|
| Read | L | H | L | H | H | X | $A_{IN}$ | $D_{OUT}$ |
| | H | L | | | | | | |
| Write | L | H | H | L | H | X (Note 2) | $A_{IN}$ | $D_{IN}$ |
| | H | L | | | | | | |
| Standby | H | H | X | X | H | X | X | High-Z |
| Output Disable | L | L | H | H | H | X | X | High-Z |
| Reset | X | X | X | X | L | X | X | High-Z |
| Temporary Sector Unprotect (High Voltage) | X | X | X | X | $V_{ID}$ | X | $A_{IN}$ | $D_{IN}$ |

***Legend***

*L = Logic Low = $V_{IL}$*
*H = Logic High = $V_{IH}$*
*VID = 11.5–12.5 V*
*$V_{HH}$ = 8.5 – 9.5 V*
*X = Don't Care*
*SA = Sector Address*
*$A_{IN}$ = Address In*
*$D_{IN}$ = Data In*
*$D_{OUT}$ = Data Out*

***Notes***

1. *The sector and sector unprotect functions may also be implemented by programming equipment.*
2. *WP#/ACC must be high when writing to the upper two and lower two sectors.*

## 7.2 Asynchronous Read

The internal state machine is set for reading array data upon device power-up, or after a hardware reset. This ensures that no spurious alteration of the memory content occurs during the power transition. No command is necessary in this mode to obtain array data. Standard microprocessor read cycles that assert valid addresses on the device address inputs produce valid data on the device data outputs. Each bank remains enabled for read access until the command register contents are altered.

### 7.2.1 Non-Page Random Read

Address access time ($t_{ACC}$) is equal to the delay from stable addresses to valid output data. The chip enable access time ($t_{CE}$) is the delay from the stable addresses and stable CE# to valid data at the output inputs. The output enable access time is the delay from the falling edge of the OE# to valid data at the output (assuming the addresses have been stable for at least $t_{ACC} - t_{OE}$ time).

### 7.2.2 Page Mode Read

The device is capable of fast page mode read and is compatible with the page mode Mask ROM read operation. This mode provides faster read access speed for random locations within a page. The random or initial page access is $t_{ACC}$ or $t_{CE}$ and subsequent page read accesses (as long as the locations specified by the microprocessor falls within that page) is equivalent to $t_{PACC}$. When CE# is deasserted (= $V_{IH}$), the reassertion of CE# for subsequent access has access time of $t_{ACC}$ or $t_{CE}$. Here again, CE# selects the device and OE# is the output control and should be used to gate data to the output inputs if the device is selected. Fast page mode accesses are obtained by keeping $A_{max}$ – A3 constant and changing A2 – A0 to select the specific word within that page.

Address bits $A_{max}$ – A3 select an 8-word page, and address bits A2 – A0 select a specific word within that page. This is an asynchronous operation with the microprocessor supplying the specific word location. See Table 7.3 for details on selecting specific words.

The device is automatically set to reading array data after device power-up. No commands are required to retrieve data. Each bank is ready to read array data after completing an Embedded Program or Embedded Erase algorithm. All addresses are latched on the falling edge of WE# or CE#, whichever happens later. All data is latched on the rising edge of WE# or CE#, whichever happens first.

Reads from the memory array may be performed in conjunction with the Erase Suspend and Program Suspend features. After the device accepts an Erase Suspend command, the corresponding bank enters the erase-suspend-read mode, after which the system can read data from any non-erase-suspended sector within the same bank. The system can read array data using the standard read timing, except that if it reads at an address within erase-suspended sectors, the device outputs status data. After completing a programming operation in the Erase Suspend mode, the system may once again read array data with the same exception. After the device accepts a Program Suspend command, the corresponding bank enters the program-suspend-read mode, after which the system can read data from any non-program-suspended sector within the same bank.

**Table 7.3** Word Selection within a Page

| Word | A2 | A1 | A0 |
|---|---|---|---|
| Word 0 | 0 | 0 | 0 |
| Word 1 | 0 | 0 | 1 |
| Word 2 | 0 | 1 | 0 |
| Word 3 | 0 | 1 | 1 |
| Word 4 | 1 | 0 | 0 |
| Word 5 | 1 | 0 | 1 |
| Word 6 | 1 | 1 | 0 |
| Word 7 | 1 | 1 | 1 |

## 7.3 Autoselect

The Autoselect mode allows the host system to access manufacturer and device identification, and verify sector protection, through identifier codes output from the internal register (separate from the memory array) on DQ15-DQ0. This mode is primarily intended to allow equipment to automatically match a device to be programmed with its corresponding programming algorithm. When verifying sector protection, the sector address must appear on the appropriate highest order address bits (see Table 7.5). The remaining address bits are don't care. When all necessary bits have been set as required, the programming equipment can then read the corresponding identifier code on DQ15-DQ0.

The Autoselect codes can also be accessed in-system through the command register. Note that if a Bank Address (BA) on the four uppermost address bits is asserted during the third write cycle of the Autoselect command, the host system can read Autoselect data from that bank and then immediately read array data from the other bank, without exiting the Autoselect mode.

- To access the Autoselect codes, the host system must issue the Autoselect command.
- The Autoselect command sequence can be written to an address within a bank that is either in the read or erase-suspend-read mode.
- The Autoselect command cannot be written while the device is actively programming or erasing in the other bank.
- Autoselect does not support simultaneous operations or page modes.
- The system must write the reset command to return to the read mode (or erase-suspend-read mode if the bank was previously in Erase Suspend).

See Table 12.1 on page 66 for command sequence details.

**Table 7.4** Autoselect Codes

| Description | | CE# (See Note) | OE# | WE# | $A_{max}$ –A12 | A10 | A9 | A8 | A7 | A6 | A5 –A4 | A3 | A2 | A1 | A0 | DQ15 to DQ0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Manufacturer ID | | L | L | H | BA | X | X | X | L | L | X | L | L | L | L | 0001h |
| Device ID: | Read Cycle 1 | L | L | H | BA | X | X | X | L | L | L | L | L | L | H | 227Eh |
| | Read Cycle 2 | L | | | | | | | | | | H | H | H | L | 223Ch (PL256N)<br>2220h (PL127N)<br>2221h (PL129N) |
| | Read Cycle 3 | L | | | | | | | | | | H | H | H | H | 2200h (PL256N)<br>2200h (PL127N)<br>2200h (PL129N) |
| Sector Protection Verification | | L | L | H | SA | X | X | X | L | L | L | L | L | H | L | 0000h<br>Unprotected (Neither DYB nor PPB Locked)<br>0001h<br>Protected (Either DYB or PPB Locked) |
| Indicator Bit | | L | L | H | BA | X | X | X | L | L | L | L | L | H | H | DQ15 - DQ8 = 0<br>DQ7 - Factory Lock Bit<br>1 = Locked<br>0 = Not Locked<br>DQ6 -Customer Lock Bit<br>1 = Locked<br>0 = Not Locked<br>DQ5 - Handshake Bit<br>1 = Reserved<br>0 = Standard Handshake<br>DQ4 & DQ3 -WP# Protection Boot Code<br>00 = WP# Protects both Top Boot and Bottom Boot Sectors<br>11 = No WP# Protection<br>DQ2 - DQ0 = 0 |

***Legend***

*L = Logic Low = $V_{IL}$  H = Logic High = $V_{IH}$*

*BA = Bank Address  SA = Sector Address  X = Don't care*

***Note***

*For the PL129N either CE1# or CE2# must be low to access Autoselect Codes*

## Software Functions and Sample Code

**Table 7.5** Autoselect Entry
(LLD Function = lld_AutoselectEntryCmd)

| Cycle | Operation | Word Address | Data |
|---|---|---|---|
| Unlock Cycle 1 | Write | BAx555h | 0x00AAh |
| Unlock Cycle 2 | Write | BAx2AAh | 0x0055h |
| Autoselect Command | Write | BAx555h | 0x0090h |

**Table 7.6** Autoselect Exit
(LLD Function = lld_AutoselectExitCmd)

| Cycle | Operation | Word Address | Data |
|---|---|---|---|
| Unlock Cycle 1 | Write | base + xxxh | 0x00F0h |

***Notes***

1. *Any offset within the device works.*
2. *BA = Bank Address. The bank address is required.*
3. *base = base address.*

The following is a C source code example of using the autoselect function to read the manufacturer ID. See the *Spansion Low Level Driver User's Guide* (available on www.spansion.com) for general information on Spansion Flash memory software development guidelines.

```
/* Here is an example of Autoselect mode (getting manufacturer ID) */
/* Define UINT16 example: typedef unsigned short UINT16; */

UINT16 manuf_id;

/* Auto Select Entry */

*((UINT16 *)bank_addr + 0x555) = 0x00AA; /* write unlock cycle 1 */
*((UINT16 *)bank_addr + 0x2AA) = 0x0055; /* write unlock cycle 2 */
*((UINT16 *)bank_addr + 0x555) = 0x0090; /* write autoselect command */


/* multiple reads can be performed after entry */

manuf_id = *((UINT16 *)bank_addr + 0x000); /* read manuf. id */

/* Autoselect exit */

*((UINT16 *)base_addr + 0x000) = 0x00F0; /* exit autoselect (write reset command) */
```

## 7.4 Program/Erase Operations

These devices are capable of single word or write buffer programming operations which are described in the following sections. The write buffer programming is recommended over single word programming as it has clear benefits from greater programming efficiency. See Table 7.1 on page 20 for the correct device settings required before initiation of a write command sequence.

Note the following details regarding the program/erase operations:

- When the Embedded Program algorithm is complete, the device then returns to the read mode.
- The system can determine the status of the program operation by using DQ7 or DQ6. See *Write Operation Status* on page 37 for information on these status bits.
- A *0* cannot be programmed back to a *1*. Attempting to do so causes the device to set DQ5 = 1 (halting any further operation and requiring a reset command). A succeeding read shows that the data is still *0*.
- Only erase operations can convert a *0* to a *1*.
- A hardware reset immediately terminates the program operation and the program command sequence should be reinitiated once the device has returned to the read mode, to ensure data integrity.
- Any commands written to the device during the Embedded Program Algorithm are ignored except the Program Suspend command.
- Secured Silicon Sector, Autoselect, and CFI functions are unavailable when a program operation is in progress.
- Programming is allowed in any sequence and across sector boundaries for single word programming operation.

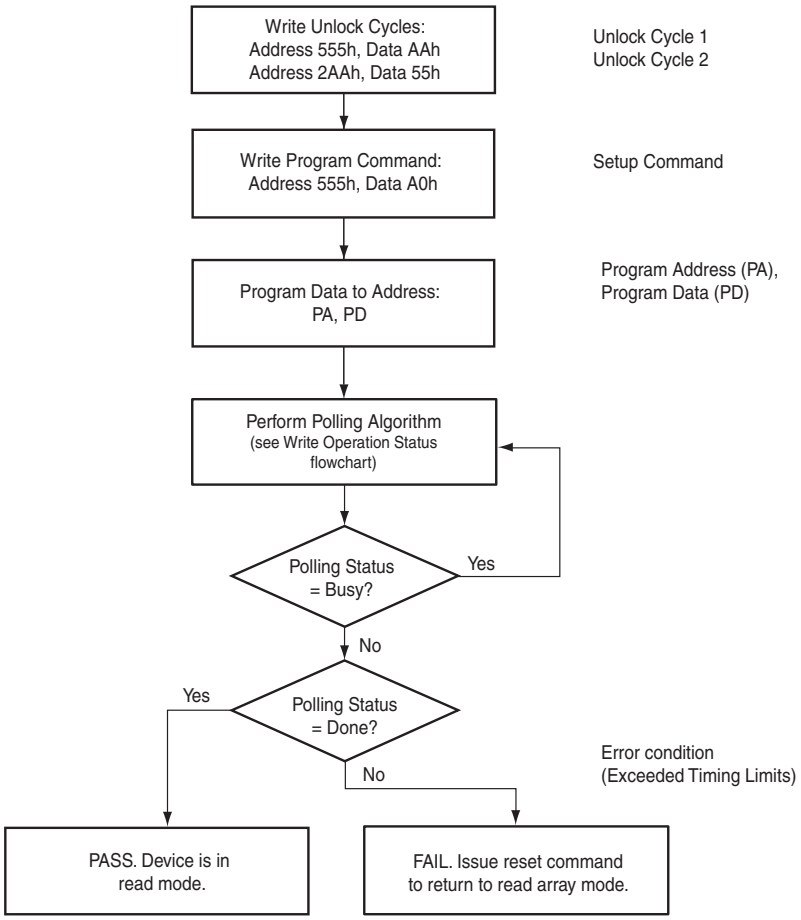
### 7.4.1 Single Word Programming

In single word programming mode, four Flash command write cycles are used to program an individual Flash address. While this method is supported by all Spansion devices, in general it is not recommended for devices that support Write Buffer Programming. See Table 12.1 on page 66 for the required bus cycles and Figure 7.1 on page 26 for the flowchart.

When the Embedded Program algorithm is complete, the device then returns to the read mode and addresses are no longer latched. The system can determine the status of the program operation by using DQ7 or DQ6. See *Write Operation Status* on page 37 for information on these status bits.

Single word programming is supported for backward compatibility with existing Flash driver software and use of write buffer programming is strongly recommended for general programming. The effective word programming time using write buffer programming is approximately four times faster than the single word programming time.

**Figure 7.1** Single Word Program Operation




## Software Functions and Sample Code

**Table 7.7** Single Word Program
(LLD Function = lld_ProgramCmd)

| Cycle | Operation | Word Address | Data |
|---|---|---|---|
| Unlock Cycle 1 | Write | Base + 555h | 00AAh |
| Unlock Cycle 2 | Write | Base + 2AAh | 0055h |
| Program Setup | Write | Base + 555h | 00A0h |
| Program | Write | Word Address | Data Word |

***Note***
*Base = Base Address.*

The following is a C source code example of using the single word program function. See the *Spansion Low Level Driver User's Guide* (available on www.spansion.com) for general information on Spansion Flash memory software development guidelines.

```
/* Example: Program Command    */
 *((UINT16 *)base_addr + 0x555) = 0x00AA;   /* write unlock cycle 1           */
 *((UINT16 *)base_addr + 0x2AA) = 0x0055;   /* write unlock cycle 2           */
 *((UINT16 *)base_addr + 0x555) = 0x00A0;   /* write program setup command    */
 *((UINT16 *)pa)                = data;     /* write data to be programmed    */
/* Poll for program completion */
```

### 7.4.2 Write Buffer Programming

Write Buffer Programming allows the system to write a maximum of 32 words in one programming operation. This results in a faster effective word programming time than the standard *word* programming algorithms. The Write Buffer Programming command sequence is initiated by first writing two unlock cycles. This is followed by a third write cycle containing the Write Buffer Load command written at the Sector Address in which programming occurs. At this point, the system writes the number of *word locations minus 1* that is loaded into the page buffer at the Sector Address in which programming occurs. This tells the device how many write buffer addresses are loaded with data and therefore when to expect the *Program Buffer to Flash* confirm command. The number of locations to program cannot exceed the size of the write buffer or the operation aborts. (Number loaded = the number of locations to program minus 1. For example, if the system programs 6 address locations, then 05h should be written to the device.)

The system then writes the starting address/data combination. This starting address is the first address/data pair to be programmed, and selects the *write-buffer-page* address. All subsequent address/data pairs must fall within the elected-write-buffer-page.

The *write-buffer-page* is selected by using the addresses $A_{max}$ – A5.

The *write-buffer-page* addresses must be the same for all address/data pairs loaded into the write buffer. (This means Write Buffer Programming cannot be performed across multiple *write-buffer-page*. This also means that Write Buffer Programming cannot be performed across multiple sectors. If the system attempts to load programming data outside of the selected *write-buffer-page*, the operation ABORTS.)

After writing the Starting Address/Data pair, the system then writes the remaining address/data pairs into the write buffer.

Note that if a Write Buffer address location is loaded multiple times, the *address/data pair* counter decrements for every data load operation. Also, the last data loaded at a location before the *Program Buffer to Flash* confirm command is programmed into the device. The software takes care of the ramifications of loading a write-buffer location more than once. The counter decrements for each data load operation, NOT for each unique write-buffer-address location. Once the specified number of write buffer locations have been loaded, the system must then write the *Program Buffer to Flash* command at the Sector Address. Any other address/data write combinations abort the Write Buffer Programming operation. The device then *goes* busy. The Data Bar polling techniques should be used while monitoring the last address location loaded into the write buffer. This eliminates the need to store an address in memory because the system can load the last address location, issue the program confirm command at the last loaded address location, and then data bar poll at that same address.

The write-buffer *embedded* programming operation can be suspended using the standard suspend/resume commands. Upon successful completion of the Write Buffer Programming operation, the device returns to READ mode.

If the write buffer command sequence is entered incorrectly the device enters write buffer abort. When an abort occurs the *write-to buffer-abort reset* command must be issued to return the device to read mode.

The Write Buffer Programming Sequence is ABORTED under any of the following conditions:

- Load a value that is greater than the page buffer size during the *Number of Locations to Program* step.
- Write to an address in a sector different than the one specified during the *Write-Buffer-Load* command.
- Write an Address/Data pair to a different write-buffer-page than the one selected by the *Starting Address* during the *write buffer data loading* stage of the operation.
- Write data other than the *Confirm Command* after the specified number of *data load* cycles.

Use of the write buffer is strongly recommended for programming when multiple words are to be programmed. Write buffer programming is approximately four times faster than programming one word at a time. Note that the Secured Silicon, the CFI functions, and the Autoselect Codes are not available for read when a write buffer programming operation is in progress.

## Software Functions and Sample Code

**Table 7.8** Write Buffer Program
(LLD Functions Used = lld_WriteToBufferCmd, lld_ProgramBufferToFlashCmd)

| Cycle | Description | Operation | Word Address | Data |
|---|---|---|---|---|
| 1 | Unlock | Write | Base + 555h | 00AAh |
| 2 | Unlock | Write | Base + 2AAh | 0055h |
| 3 | Write Buffer Load Command | Write | Program Address | 0025h |
| 4 | Write Word Count | Write | Program Address | Word Count (N–1)h |
| **Number of words (N) loaded into the write buffer can be from 1 to 32 words.** | | | | |
| 5 to 36 | Load Buffer Word N | Write | Program Address, Word N | Word N |
| Last | Write Buffer to Flash | Write | Sector Address | 0029h |

***Notes***

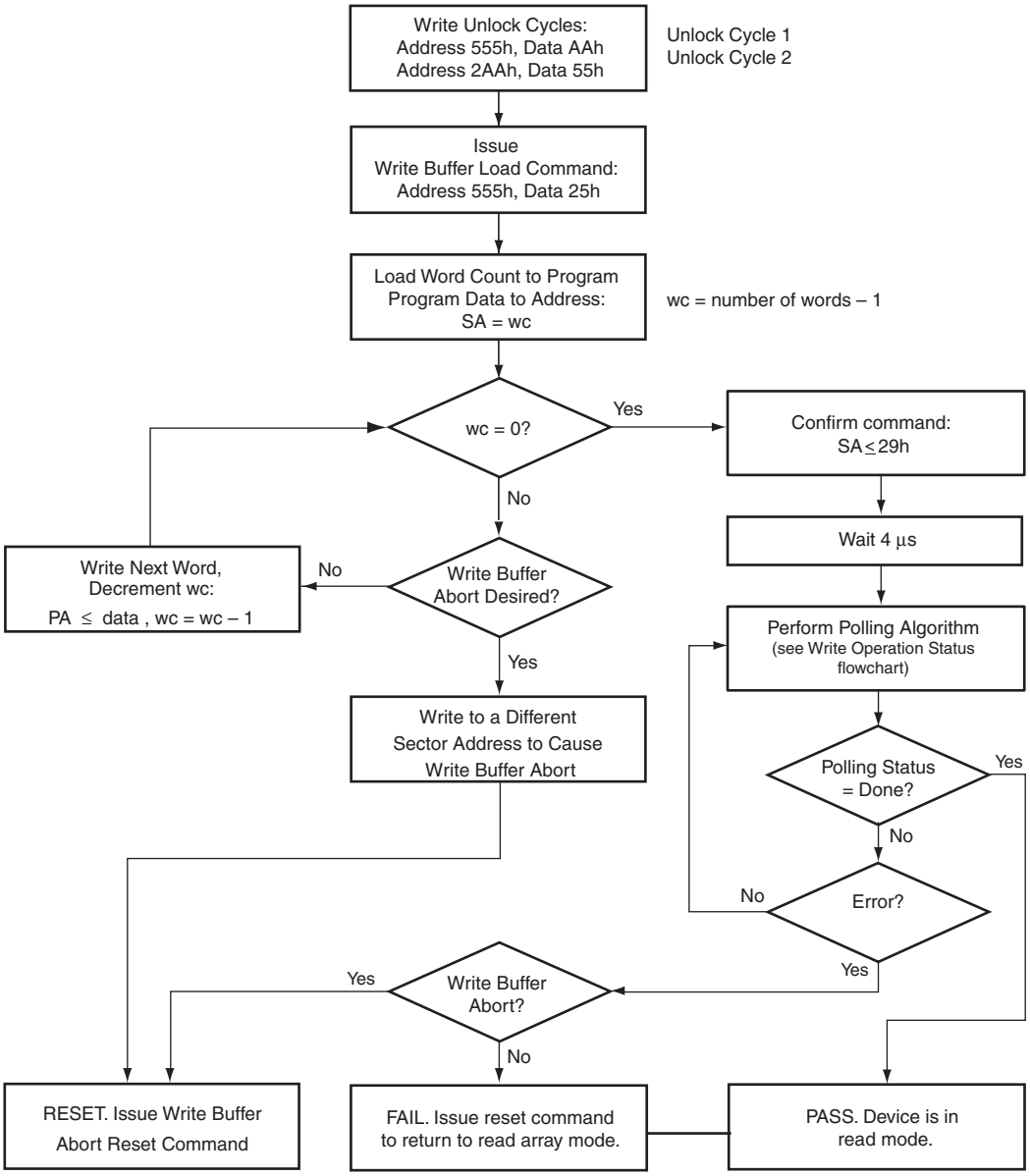
1. *Base = Base Address.*
2. *Last = Last cycle of write buffer program operation; depending on number of words written, the total number of cycles can be from 6 to 37.*
3. *For maximum efficiency, it is recommended that the write buffer be loaded with the highest number of words (N words) possible.*

The following is a C source code example of using the write buffer program function. See the *Spansion Low Level Driver User's Guide* (available on www.spansion.com) for general information on Spansion Flash memory software development guidelines.

```
/* Example: Write Buffer Programming Command            */
/* NOTES: Write buffer programming limited to 16 words. */
/*        All addresses to be written to the flash in   */
/*        one operation must be within the same flash   */
/*        page. A flash page begins at addresses       */
/*        evenly divisible by 0x20.                     */
 UINT16 *src = source_of_data;               /* address of source data          */
 UINT16 *dst = destination_of_data;          /* flash destination address       */
 UINT16 wc   = words_to_program -1;          /* word count (minus 1)            */
 *((UINT16 *)base_addr + 0x555) = 0x00AA;   /* write unlock cycle 1            */
 *((UINT16 *)base_addr + 0x2AA) = 0x0055;   /* write unlock cycle 2            */
 *((UINT16 *)sector_address)    = 0x0025;   /* write write buffer load command */
 *((UINT16 *)sector_address)    = wc;       /* write word count (minus 1)      */
loop:
 *dst = *src; /* ALL dst MUST BE SAME PAGE */ /* write source data to destination */
 dst++;                                      /* increment destination pointer    */
 src++;                                      /* increment source pointer         */
 if (wc == 0) goto confirm                   /* done when word count equals zero */
 wc--;                                       /* decrement word count             */
 goto loop;                                  /* do it again                      */
confirm:
 *((UINT16 *)sector_address)    = 0x0029;   /* write confirm command           */
 /* poll for completion */


/* Example: Write Buffer Abort Reset */
 *((UINT16 *)addr + 0x555) = 0x00AA;   /* write unlock cycle 1            */
 *((UINT16 *)addr + 0x2AA) = 0x0055;   /* write unlock cycle 2            */
 *((UINT16 *)addr + 0x555) = 0x00F0;   /* write buffer abort reset        */
```

**Figure 7.2** Write Buffer Programming Operation



### 7.4.3 Sector Erase

The sector erase function erases one or more sectors in the memory array. (See Table 12.1 on page 66, and Figure 7.3 on page 31.) The device does not require the system to preprogram prior to erase. The Embedded Erase algorithm automatically programs and verifies the entire memory for an all zero data pattern prior to electrical erase. The system is not required to provide any controls or timings during these operations.

After the command sequence is written, a sector erase time-out of no less than $t_{SEA}$ occurs. During the time-out period, additional sector addresses and sector erase commands can be written. Loading the sector erase buffer can be done in any sequence, and the number of sectors can be from one sector to all sectors. The time between these additional cycles must be less than $t_{SEA}$. Any sector erase address and command following the exceeded time-out ($t_{SEA}$) may or may not be accepted. Any command other than Sector Erase or Erase Suspend during the time-out period resets that bank to the read mode. The system can monitor DQ3 to determine if the sector erase timer has timed out (see *DQ3: Sector Erase Timeout State Indicator* on page 40). The time-out begins from the rising edge of the final WE# pulse in the command sequence.

When the Embedded Erase algorithm is complete, the bank returns to reading array data and addresses are no longer latched. Note that while the Embedded Erase operation is in progress, the system can read data from the non-erasing banks. The system can determine the status of the erase operation by reading DQ7 or DQ6/DQ2 in the erasing bank. See *Write Operation Status* on page 37 for information on these status bits.

Once the sector erase operation has begun, only the Erase Suspend command is valid. All other commands are ignored. However, note that a hardware reset immediately terminates the erase operation. If that occurs, the sector erase command sequence should be reinitiated once that bank has returned to reading array data, to ensure data integrity.

Figure 7.3 on page 31 illustrates the algorithm for the erase operation. See *AC Characteristics* on page 59 for the Erase/Program Operations parameters and timing diagrams.

#### Software Functions and Sample Code

**Table 7.9** Sector Erase
(LLD Function = lld_SectorEraseCmd)

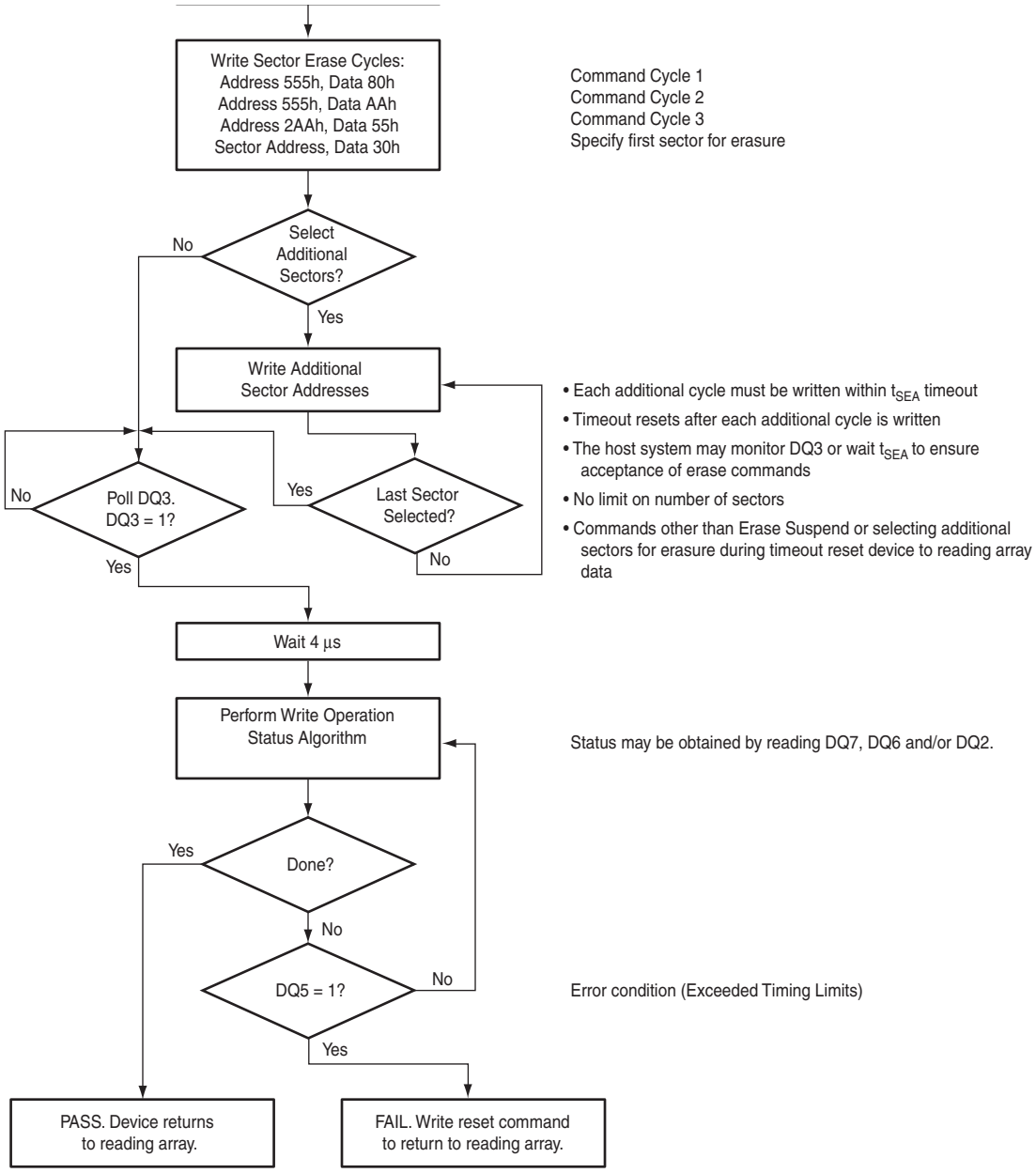
| Cycle | Description | Operation | Word Address | Data |
|---|---|---|---|---|
| 1 | Unlock | Write | Base + 555h | 00AAh |
| 2 | Unlock | Write | Base + 2AAh | 0055h |
| 3 | Setup Command | Write | Base + 555h | 0080h |
| 4 | Unlock | Write | Base + 555h | 00AAh |
| 5 | Unlock | Write | Base + 2AAh | 0055h |
| 6 | Sector Erase Command | Write | Sector Address | 0030h |

***Note***
*Unlimited additional sectors can be selected for erase; command(s) must be written within $t_{SEA}$.*

The following is a C source code example of using the sector erase function. Refer to the *Spansion Low Level Driver User's Guide* (available on www.spansion.com) for general information on Spansion Flash memory software development guidelines.

```
/* Example: Sector Erase Command */
 *((UINT16 *)base_addr + 0x555) = 0x00AA;   /* write unlock cycle 1            */
 *((UINT16 *)base_addr + 0x2AA) = 0x0055;   /* write unlock cycle 2            */
 *((UINT16 *)base_addr + 0x555) = 0x0080;   /* write setup command             */
 *((UINT16 *)base_addr + 0x555) = 0x00AA;   /* write additional unlock cycle 1 */
 *((UINT16 *)base_addr + 0x2AA) = 0x0055;   /* write additional unlock cycle 2 */
 *((UINT16 *)sector_address)    = 0x0030;   /* write sector erase command      */
```

**Figure 7.3** Sector Erase Operation




***Notes***

1. *See Table 12.1 on page 66 for erase command sequence.*
2. *See the section on DQ3 for information on the sector erase timeout.*

### 7.4.4 Chip Erase Command Sequence

Chip erase is a six-bus cycle operation as indicated by Table 12.1 on page 66. These commands invoke the Embedded Erase algorithm, which does not require the system to preprogram prior to erase. The Embedded Erase algorithm automatically preprograms and verifies the entire memory for an all zero data pattern prior to electrical erase. The system is not required to provide any controls or timings during these operations. The Command Definition tables (Table 12.1 on page 66 and Table 12.2 on page 68) show the address and data requirements for the chip erase command sequence.

When the Embedded Erase algorithm is complete, that bank returns to the read mode and addresses are no longer latched. The system can determine the status of the erase operation by using DQ7 or DQ6/DQ2. See *Write Operation Status* on page 37 for information on these status bits.

Any commands written during the chip erase operation are ignored. However, note that a hardware reset immediately terminates the erase operation. If that occurs, the chip erase command sequence should be reinitiated once that bank has returned to reading array data, to ensure data integrity.

#### Software Functions and Sample Code

**Table 7.10** Chip Erase
(LLD Function = lld_ChipEraseCmd)

| Cycle | Description | Operation | Word Address | Data |
|---|---|---|---|---|
| 1 | Unlock | Write | Base + 555h | 00AAh |
| 2 | Unlock | Write | Base + 2AAh | 0055h |
| 3 | Setup Command | Write | Base + 555h | 0080h |
| 4 | Unlock | Write | Base + 555h | 00AAh |
| 5 | Unlock | Write | Base + 2AAh | 0055h |
| 6 | Chip Erase Command | Write | Base + 555h | 0010h |

The following is a C source code example of using the chip erase function. Refer to the *Spansion Low Level Driver User's Guide* (available on www.spansion.com) for general information on Spansion Flash memory software development guidelines.

```
/* Example: Chip Erase Command */
/* Note: Cannot be suspended   */
 *((UINT16 *)base_addr + 0x555) = 0x00AA;   /* write unlock cycle 1            */
 *((UINT16 *)base_addr + 0x2AA) = 0x0055;   /* write unlock cycle 2            */
 *((UINT16 *)base_addr + 0x555) = 0x0080;   /* write setup command             */
 *((UINT16 *)base_addr + 0x555) = 0x00AA;   /* write additional unlock cycle 1 */
 *((UINT16 *)base_addr + 0x2AA) = 0x0055;   /* write additional unlock cycle 2 */
 *((UINT16 *)base_addr + 0x000) = 0x0010;   /* write chip erase command        */
```

### 7.4.5 Erase Suspend/Erase Resume Commands

The Erase Suspend command allows the system to interrupt a sector erase operation and then read data from, or program data to, any sector not selected for erasure. The bank address is required when writing this command. This command is valid only during the sector erase operation, including the minimum $t_{SEA}$ time-out period during the sector erase command sequence. The Erase Suspend command is ignored if written during the chip erase operation.

When the Erase Suspend command is written during the sector erase operation, the device requires a maximum of $t_{ESL}$ (erase suspend latency) to suspend the erase operation. However, when the Erase Suspend command is written during the sector erase time-out, the device immediately terminates the time-out period and suspends the erase operation.

After the erase operation has been suspended, the bank enters the erase-suspend-read mode. The system can read data from or program data to any sector not selected for erasure. (The device *erase suspends* all sectors selected for erasure.) Reading at any address within erase-suspended sectors produces status information on DQ7-DQ0. The system can use DQ7, or DQ6, and DQ2 together, to determine if a sector is actively erasing or is erase-suspended. Refer to Table 7.18 on page 40 for information on these status bits.

After an erase-suspended program operation is complete, the bank returns to the erase-suspend-read mode. The system can determine the status of the program operation using the DQ7 or DQ6 status bits, just as in the standard program operation.

In the erase-suspend-read mode, the system can also issue the Autoselect command sequence. See *Write Buffer Programming* on page 27 and *Autoselect* on page 23 for details.

To resume the sector erase operation, the system must write the Erase Resume command. The bank address of the erase-suspended bank is required when writing this command. Further writes of the Resume command are ignored. Another Erase Suspend command can be written after the chip has resumed erasing.

#### Software Functions and Sample Code

**Table 7.11** Erase Suspend
(LLD Function = lld_EraseSuspendCmd)

| Cycle | Operation | Word Address | Data |
|---|---|---|---|
| 1 | Write | Bank Address | 00B0h |

The following is a C source code example of using the erase suspend function. Refer to the *Spansion Low Level Driver User's Guide* (available on www.spansion.com) for general information on Spansion Flash memory software development guidelines.

```
/* Example: Erase suspend command */
 *((UINT16 *)bank_addr + 0x000) = 0x00B0;   /* write suspend command          */
```

**Table 7.12** Erase Resume
(LLD Function = lld_EraseResumeCmd)

| Cycle | Operation | Word Address | Data |
|---|---|---|---|
| 1 | Write | Bank Address | 0030h |

The following is a C source code example of using the erase resume function. Refer to the *Spansion Low Level Driver User's Guide* (available on www.spansion.com) for general information on Spansion Flash memory software development guidelines.

```
/* Example: Erase resume command */
 *((UINT16 *)bank_addr + 0x000) = 0x0030;   /* write resume command           */
 /* The flash needs adequate time in the resume state */
```

### 7.4.6 Program Suspend/Program Resume Commands

The Program Suspend command allows the system to interrupt an embedded programming operation or a *Write to Buffer* programming operation so that data can read from any non-suspended sector. When the Program Suspend command is written during a programming process, the device halts the programming operation within $t_{PSL}$ (program suspend latency) and updates the status bits.

After the programming operation has been suspended, the system can read array data from any non-suspended sector. The Program Suspend command can also be issued during a programming operation while an erase is suspended. In this case, data can be read from any addresses not in Erase Suspend or Program Suspend. If a read is needed from the Secured Silicon Sector area, then user must use the proper command sequences to enter and exit this region.

The system can also write the Autoselect command sequence when the device is in Program Suspend mode. The device allows reading Autoselect codes in the suspended sectors, since the codes are not stored in the memory array. When the device exits the Autoselect mode, the device reverts to Program Suspend mode, and is ready for another valid operation. See *Autoselect* on page 23 for more information.

After the Program Resume command is written, the device reverts to programming. The system can determine the status of the program operation using the DQ7 or DQ6 status bits, just as in the standard program operation. See *Write Operation Status* on page 37 for more information.

The system must write the Program Resume command (address bits are *don't cares*) to exit the Program Suspend mode and continue the programming operation. Further writes of the Program Resume command are ignored. Another Program Suspend command can be written after the device has resumed programming.

#### Software Functions and Sample Code

**Table 7.13** Program Suspend
(LLD Function = lld_ProgramSuspendCmd)

| Cycle | Operation | Word Address | Data |
|---|---|---|---|
| 1 | Write | Bank Address | 00B0h |

The following is a C source code example of using the program suspend function. Refer to the *Spansion Low Level Driver User's Guide* (available on www.spansion.com) for general information on Spansion Flash memory software development guidelines.

```
/* Example: Program suspend command */
 *((UINT16 *)base_addr + 0x000) = 0x00B0;   /* write suspend command          */
```

**Table 7.14** Program Resume
(LLD Function = lld_ProgramResumeCmd)

| Cycle | Operation | Word Address | Data |
|---|---|---|---|
| 1 | Write | Bank Address | 0030h |

The following is a C source code example of using the program resume function. Refer to the *Spansion Low Level Driver User's Guide* (available on www.spansion.com) for general information on Spansion Flash memory software development guidelines.

```
/* Example: Program resume command */
 *((UINT16 *)base_addr + 0x000) = 0x0030;   /* write resume command           */
```

### 7.4.7 Accelerated Program

Accelerated single word programming, write buffer programming, sector erase, and chip erase operations are enabled through the ACC function. This method is faster than the standard chip program and erase command sequences.

**The accelerated chip program and erase functions must not be used more than 10 times per sector.** In addition, accelerated chip program and erase should be performed at room temperature (25°C ±10°C).

This function is primarily intended to allow faster manufacturing throughput at the factory. If the system asserts $V_{HH}$ on this input, the device automatically enters the aforementioned Unlock Bypass mode and uses the higher voltage on the input to reduce the time required for program and erase operations. The system can then use the Write Buffer Load command sequence provided by the Unlock Bypass mode. Note that if a *Write-to-Buffer-Abort Reset* is required while in Unlock Bypass mode, the full 3-cycle RESET command sequence must be used to reset the device. Removing $V_{HH}$ from the ACC input, upon completion of the embedded program or erase operation, returns the device to normal operation.

- Sectors must be unlocked prior to raising WP#/ACC to $V_{HH}$.
- The WP#/ACC must not be at $V_{HH}$ for operations other than accelerated programming and accelerated chip erase, or device damage can result.
- Set the ACC pin at $V_{CC}$ when accelerated programming not in use.

### 7.4.8 Unlock Bypass

The device features an Unlock Bypass mode to facilitate faster word programming. Once the device enters the Unlock Bypass mode, only two write cycles are required to program data, instead of the normal four cycles.

This mode dispenses with the initial two unlock cycles required in the standard program command sequence, resulting in faster total programming time. Table 12.1, *Memory Array Commands* on page 66 shows the requirements for the unlock bypass command sequences.

During the unlock bypass mode, only the Read, Unlock Bypass Program and Unlock Bypass Reset commands are valid. To exit the unlock bypass mode, the system must issue the two-cycle unlock bypass reset command sequence. The first cycle must contain the bank address and the data 90h. The second cycle need only contain the data 00h. The bank then returns to the read mode.

## Software Functions and Sample Code

The following are C source code examples of using the unlock bypass entry, program, and exit functions. Refer to the *Spansion Low Level Driver User's Guide* (available soon on www.spansion.com) for general information on Spansion Flash memory software development guidelines.

**Table 7.15** Unlock Bypass Entry
(LLD Function = lld_UnlockBypassEntryCmd)

| Cycle | Description | Operation | Word Address | Data |
|---|---|---|---|---|
| 1 | Unlock | Write | Base + 555h | 00AAh |
| 2 | Unlock | Write | Base + 2AAh | 0055h |
| 3 | Entry Command | Write | Base + 555h | 0020h |

```
/* Example: Unlock Bypass Entry Command    */
 *((UINT16 *)bank_addr + 0x555) = 0x00AA;   /* write unlock cycle 1            */
 *((UINT16 *)bank_addr + 0x2AA) = 0x0055;   /* write unlock cycle 2            */
 *((UINT16 *)bank_addr + 0x555) = 0x0020;   /* write unlock bypass command     */
 /* At this point, programming only takes two write cycles.   */
 /* Once you enter Unlock Bypass Mode, do a series of like    */
 /* operations (programming or sector erase) and then exit    */
 /* Unlock Bypass Mode before beginning a different type of   */
 /* operations.                                               */
```

**Table 7.16** Unlock Bypass Program
(LLD Function = lld_UnlockBypassProgramCmd)

| Cycle | Description | Operation | Word Address | Data |
|---|---|---|---|---|
| 1 | Program Setup Command | Write | Base +xxxh | 00A0h |
| 2 | Program Command | Write | Program Address | Program Data |

```
/* Example: Unlock Bypass Program Command */
/* Do while in Unlock Bypass Entry Mode!   */
 *((UINT16 *)bank_addr + 0x555) = 0x00A0;   /* write program setup command     */
 *((UINT16 *)pa)               = data;     /* write data to be programmed     */
 /* Poll until done or error.    */
 /* If done and more to program, */
 /* do above two cycles again.   */
```

**Table 7.17** Unlock Bypass Reset
(LLD Function = lld_UnlockBypassResetCmd)

| Cycle | Description | Operation | Word Address | Data |
|---|---|---|---|---|
| 1 | Reset Cycle 1 | Write | Base +xxxh | 0090h |
| 2 | Reset Cycle 2 | Write | Base +xxxh | 0000h |

```
/* Example: Unlock Bypass Exit Command */
  *( (UINT16 *)base_addr + 0x000 ) = 0x0090;
  *( (UINT16 *)base_addr + 0x000 ) = 0x0000;
```

### 7.4.9 Write Operation Status

The device provides several bits to determine the status of a program or erase operation. The following subsections describe the function of DQ1, DQ2, DQ3, DQ5, DQ6, and DQ7.

**DQ7: Data# Polling.**

The Data# Polling bit, DQ7, indicates to the host system whether an Embedded Program or Erase algorithm is in progress or completed, or whether a bank is in Erase Suspend. Data# Polling is valid after the rising edge of the final WE# pulse in the command sequence. Note that the Data# Polling is valid only for the last word being programmed in the write-buffer-page during Write Buffer Programming. Reading Data# Polling status on any word other than the last word to be programmed in the write-buffer-page returns false status information.

During the Embedded Program algorithm, the device outputs on DQ7 the complement of the datum programmed to DQ7. This DQ7 status also applies to programming during Erase Suspend. When the Embedded Program algorithm is complete, the device outputs the datum programmed to DQ7. The system must provide the program address to read valid status information on DQ7. If a program address falls within a protected sector, Data# polling on DQ7 is active for approximately $t_{PSP}$, then that bank returns to the read mode.

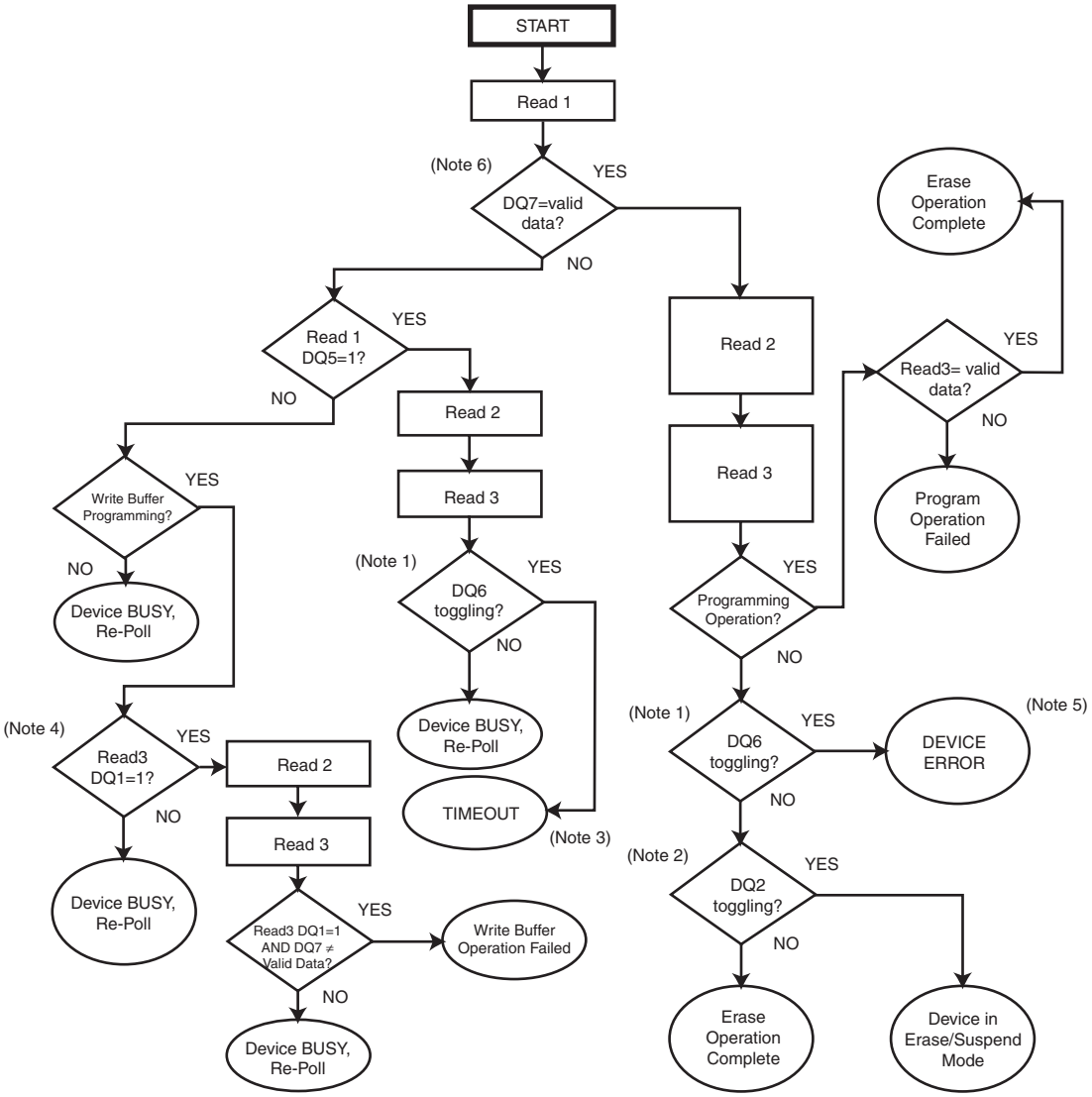
During the Embedded Erase Algorithm, Data# polling produces a *0* on DQ7. When the Embedded Erase algorithm is complete, or if the bank enters the Erase Suspend mode, Data# Polling produces a *1* on DQ7. The system must provide an address within any of the sectors selected for erasure to read valid status information on DQ7.

After an erase command sequence is written, if all sectors selected for erasing are protected, Data# Polling on DQ7 is active for approximately $t_{ASP}$, then the bank returns to the read mode. If not all selected sectors are protected, the Embedded Erase algorithm erases the unprotected sectors, and ignores the selected sectors that are protected. However, if the system reads DQ7 at an address within a protected sector, the status may not be valid.

Just prior to the completion of an Embedded Program or Erase operation, DQ7 can change asynchronously with DQ6 – DQ0 while Output Enable (OE#) is asserted low. That is, the device may change from providing status information to valid data on DQ7. Depending on when the system samples the DQ7 output, it may read the status or valid data. Even if the device has completed the program or erase operation and DQ7 has valid data, the data outputs on DQ6 – DQ0 may be still invalid. Valid data on DQ7 – DQ0 appears on successive read cycles.

See the following for more information: Table 7.18, *Write Operation Status* on page 40, shows the outputs for Data# Polling on DQ7. Figure 7.4, *Write Operation Status Flowchart* on page 38, shows the Data# Polling algorithm. Figure 11.13, *Data# Polling Timings (During Embedded Algorithms)* on page 64 shows the Data# Polling timing diagram.

**Figure 7.4** Write Operation Status Flowchart




***Notes***

1. *DQ6 is toggling if Read2 DQ6 does not equal Read3 DQ6.*
2. *DQ2 is toggling if Read2 DQ2 does not equal Read3 DQ2.*
3. *May be due to an attempt to program a 0 to 1. Use the RESET command to exit operation.*
4. *Write buffer error if DQ1 of last read =1.*
5. *Invalid state, use RESET command to exit operation.*
6. *Valid data is the data that is intended to be programmed or all 1's for an erase operation.*
7. *Data polling algorithm valid for all operations except advanced sector protection.*

### DQ6: Toggle Bit I

Toggle Bit I on DQ6 indicates whether an Embedded Program or Erase algorithm is in progress or complete, or whether the device has entered the Erase Suspend mode. Toggle Bit I can be read at any address in the same bank, and is valid after the rising edge of the final WE# pulse in the command sequence (prior to the program or erase operation), and during the sector erase time-out.

During an Embedded Program or Erase algorithm operation, successive read cycles to any address cause DQ6 to toggle. When the operation is complete, DQ6 stops toggling.

After an erase command sequence is written, if all sectors selected for erasing are protected, DQ6 toggles for approximately $t_{ASP}$ (all sectors protected toggle time), then returns to reading array data. If not all selected sectors are protected, the Embedded Erase algorithm erases the unprotected sectors, and ignores the selected sectors that are protected.

The system can use DQ6 and DQ2 together to determine whether a sector is actively erasing or is erase-suspended. When the device is actively erasing (that is, the Embedded Erase algorithm is in progress), DQ6 toggles. When the device enters the Erase Suspend mode, DQ6 stops toggling. However, the system must also use DQ2 to determine which sectors are erasing or erase-suspended. Alternatively, the system can use DQ7, see *DQ7: Data# Polling.* on page 37

If a program address falls within a protected sector, DQ6 toggles for approximately $t_{PAP}$ after the program command sequence is written, then returns to reading array data.

DQ6 also toggles during the erase-suspend-program mode, and stops toggling once the Embedded Program Algorithm is complete.

See the following for additional information: Figure 7.4, *Write Operation Status Flowchart* on page 38, Figure 11.14, *Toggle Bit Timings (During Embedded Algorithms)* on page 64, Table 7.18, *Write Operation Status* on page 40, and Figure 11.15, *DQ2 vs. DQ6* on page 64.

Toggle Bit I on DQ6 requires either OE# or CE# to be de-asserted and reasserted to show the change in state.

### DQ2: Toggle Bit II

The *Toggle Bit II* on DQ2, when used with DQ6, indicates whether a particular sector is actively erasing (that is, the Embedded Erase algorithm is in progress), or whether that sector is erase-suspended. Toggle Bit II is valid after the rising edge of the final WE# pulse in the command sequence. DQ2 toggles when the system reads at addresses within those sectors that have been selected for erasure. But DQ2 cannot distinguish whether the sector is actively erasing or is erase-suspended. DQ6, by comparison, indicates whether the device is actively erasing, or is in Erase Suspend, but cannot distinguish which sectors are selected for erasure. Thus, both status bits are required for sector and mode information. Refer to Table 7.18 on page 40 to compare outputs for DQ2 and DQ6. See the following for additional information: Figure 7.4, *Write Operation Status Flowchart* on page 38 and Figure 11.14, *Toggle Bit Timings (During Embedded Algorithms)* on page 64.

### Reading Toggle Bits DQ6/DQ2

Whenever the system initially begins reading toggle bit status, it must read DQ7 – DQ0 at least twice in a row to determine whether a toggle bit is toggling. Typically, the system would note and store the value of the toggle bit after the first read. After the second read, the system would compare the new value of the toggle bit with the first. If the toggle bit is not toggling, the device has completed the program or erases operation. The system can read array data on DQ7 – DQ0 on the following read cycle. However, if after the initial two read cycles, the system determines that the toggle bit is still toggling, the system also should note whether the value of DQ5 is high (see the section on DQ5). If it is, the system should then determine again whether the toggle bit is toggling, since the toggle bit might have stopped toggling just as DQ5 went high. If the toggle bit is no longer toggling, the device has successfully completed the program or erases operation. If it is still toggling, the device did not complete the operation successfully, and the system must write the reset command to return to reading array data. The remaining scenario is that the system initially determines that the toggle bit is toggling and DQ5 has not gone high. The system may continue to monitor the toggle bit and DQ5 through successive read cycles, determining the status as described in the previous paragraph. Alternatively, it can choose to perform other system tasks. In this case, the system must start at the beginning of the algorithm when it returns to determine the status of the operation. Refer to Figure 7.4, *Write Operation Status Flowchart* on page 38 for more details.

### DQ5: Exceeded Timing Limits

DQ5 indicates whether the program or erase time has exceeded a specified internal pulse count limit. Under these conditions DQ5 produces a *1*, indicating that the program or erase cycle was not successfully completed. The device may output a *1* on DQ5 if the system tries to program a *1* to a location that was previously programmed to *0*. Only an erase operation can change a *0* back to a *1*, Under this condition, the device halts the operation, and when the timing limit has been exceeded, DQ5 produces a *1*. Under both these conditions, the system must write the reset command to return to the read mode (or to the erase-suspend-read mode if a bank was previously in the erase-suspend-program mode).

### DQ3: Sector Erase Timeout State Indicator

After writing a sector erase command sequence, the system may read DQ3 to determine whether or not erasure has begun. (The sector erase timer does not apply to the chip erase command.) If additional sectors are selected for erasure, the entire time-out also applies after each additional sector erase command. When the time-out period is complete, DQ3 switches from a *0* to a *1*. If the time between additional sector erase commands from the system can be assumed to be less than $t_{SEA}$, the system need not monitor DQ3. See Sector Erase Command Sequence for more details.

After the sector erase command is written, the system should read the status of DQ7 (Data# Polling) or DQ6 (Toggle Bit I) to ensure that the device has accepted the command sequence, and then read DQ3. If DQ3 is *1*, the Embedded Erase algorithm has begun; all further commands (except Erase Suspend) are ignored until the erase operation is complete. If DQ3 is *0*, the device accepts additional sector erase commands. To ensure the command has been accepted, the system software should check the status of DQ3 prior to and following each sub-sequent sector erase command. If DQ3 is high on the second status check, the last command might not have been accepted. Table 7.18 shows the status of DQ3 relative to the other status bits.

### DQ1: Write to Buffer Abort

DQ1 indicates whether a Write to Buffer operation was aborted. Under these conditions DQ1 produces a *1*. The system must issue the Write to Buffer Abort Reset command sequence to return the device to reading array data. See Write Buffer Programming Operation for more details.

**Table 7.18** Write Operation Status

| Status | | | DQ7 (Note 2) | DQ6 | DQ5 (Note 1) | DQ3 | DQ2 (Note 2) | DQ1 (Note 4) |
|---|---|---|---|---|---|---|---|---|
| Program Suspend Mode (Note 3) | Reading within Program Suspended Sector | | INVALID (Not Allowed) | INVALID (Not Allowed) | INVALID (Not Allowed) | INVALID (Not Allowed) | INVALID (Not Allowed) | INVALID (Not Allowed) |
| | Reading within Non-Program Suspended Sector | | Data | Data | Data | Data | Data | Data |
| Erase Suspend Mode | Erase-Suspend-Read | Erase Suspended Sector | 1 | No Toggle | 0 | N/A | Toggle | N/A |
| | | Non-Erase Suspended Sector | Data | Data | Data | Data | Data | Data |
| | Erase-Suspend-Program | | DQ7# | Toggle | 0 | N/A | N/A | N/A |
| Write to Buffer (Note 5) | BUSY State | | DQ7# | Toggle | 0 | N/A | N/A | 0 |
| | Exceeded Timing Limits | | DQ7# | Toggle | 1 | N/A | N/A | 0 |
| | ABORT State | | DQ7# | Toggle | 0 | N/A | N/A | 1 |

***Notes***

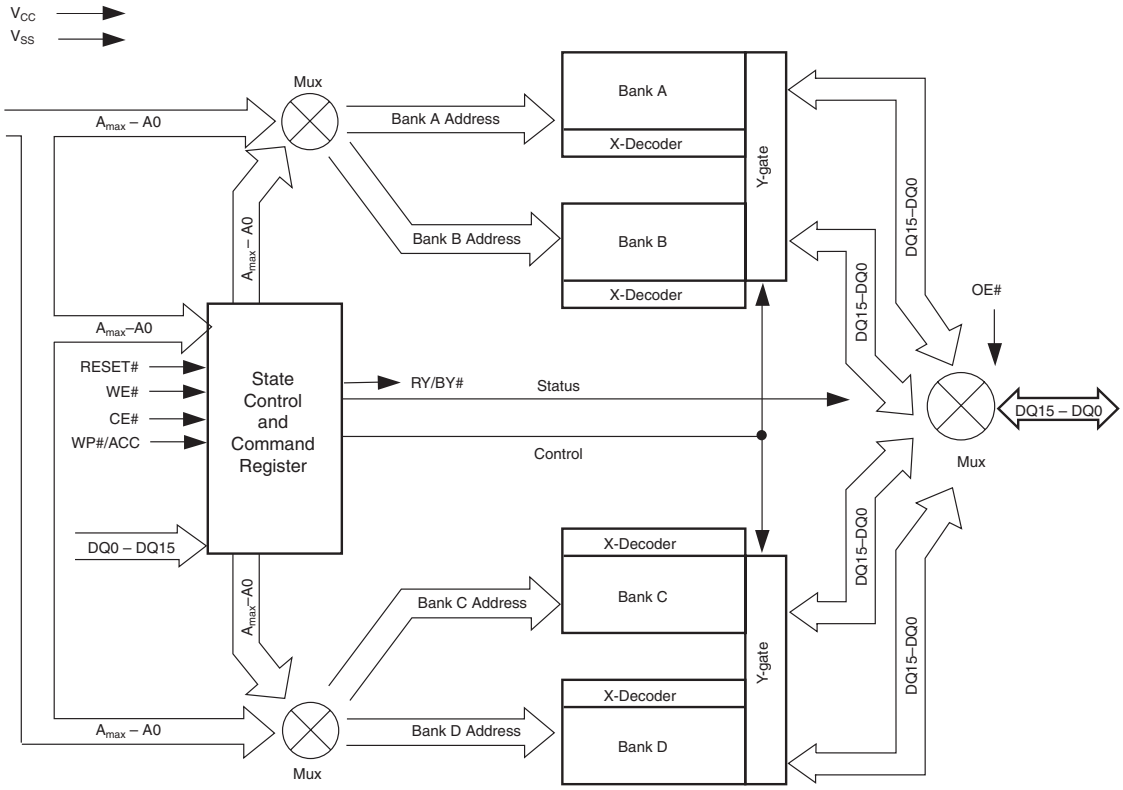
1. *DQ5 switches to '1' when an Embedded Program or Embedded Erase operation has exceeded the maximum timing limits. Refer to the section on DQ5 for more information.*
2. *DQ7 a valid address when reading status information. Refer to the appropriate subsection for further details.*
3. *Data are invalid for addresses in a Program Suspended sector.*
4. *DQ1 indicates the Write to Buffer ABORT status during Write Buffer Programming operations.*
5. *The data-bar polling algorithm should be used for Write Buffer Programming operations. Note that DQ7# during Write Buffer Programming indicates the data-bar for DQ7 data for the LAST LOADED WRITE-BUFFER ADDRESS location.*

## 7.5 Simultaneous Read/Write

The simultaneous read/write feature allows the host system to read data from one bank of memory while programming or erasing another bank of memory. An erase operation may also be suspended to read from or program another location within the same bank (except the sector being erased). Figure 11.12, *Back-to-back Read/Write Cycle Timings* on page 63 shows how read and write cycles may be initiated for simultaneous

operation with zero latency. See the table, *DC Characteristics* on page 57 for read-while-program and read-while-erase current specifications.

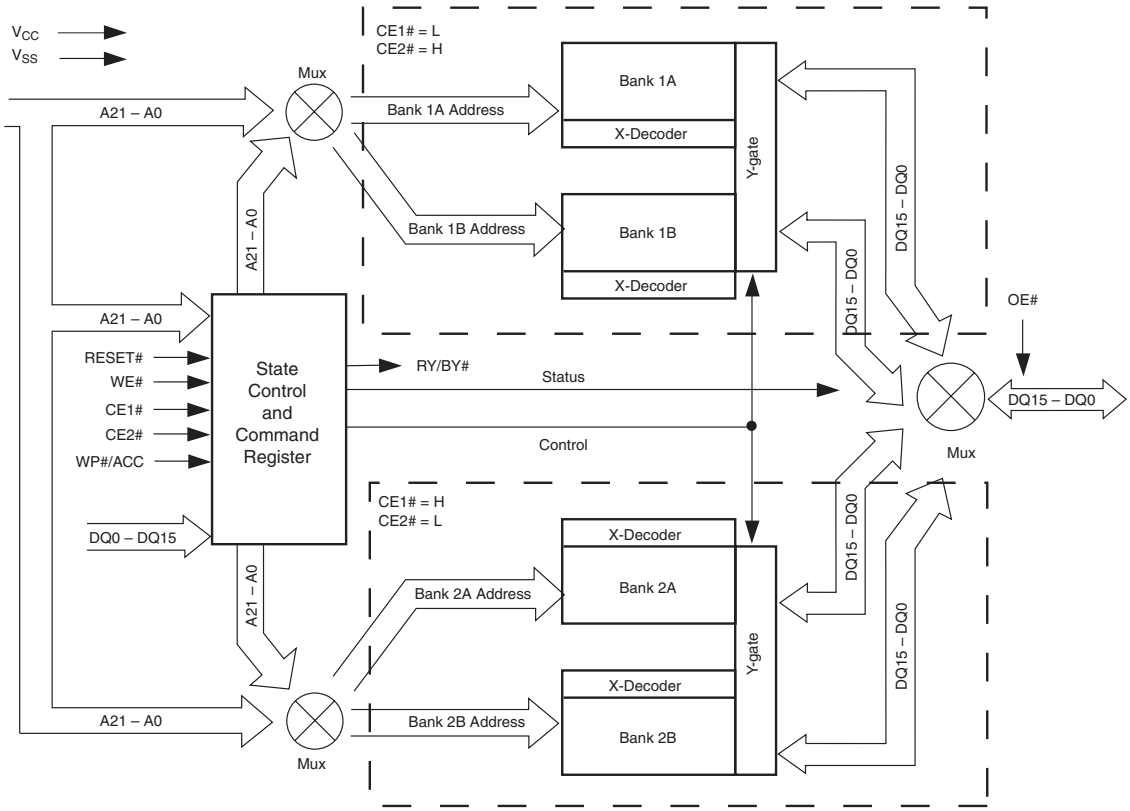
**Figure 7.5** Simultaneous Operation Block Diagram for S29PL256N and S29PL127N




***Note***

*$A_{max}$ = A23 (PL256N), A22 (PL127N)*

**Figure 7.6** Simultaneous Operation Block Diagram for S29PL129N




## 7.6 Writing Commands/Command Sequences

During a write operation, the system must drive CE# and WE# to $V_{IL}$ and OE# to $V_{IH}$ when providing an address, command, and data. Addresses are latched on the last falling edge of WE# or CE#, while data is latched on the 1st rising edge of WE# or CE#. An erase operation can erase one sector, multiple sectors, or the entire device. Table 6.1 on page 18 and Table 6.2 on page 19 indicate the address space that each sector occupies. The device address space is divided into four banks: Banks B and C contain only 128 Kword sectors, while Banks A and D contain both 32 Kword boot sectors in addition to 128 Kword sectors. A *bank address* is the set of address bits required to uniquely select a bank. Similarly, a *sector address* is the address bits required to uniquely select a sector. $I_{CC2}$ in *DC Characteristics* on page 57 represents the active current specification for the write mode. See *AC Characteristics* on page 59 contains timing specification tables and timing diagrams for write operations.

## 7.7 Hardware Reset

The RESET# input provides a hardware method of resetting the device to reading array data. When RESET# is driven low for at least a period of $t_{RP}$, the device immediately terminates any operation in progress, tristates all outputs, and ignores all read/write commands for the duration of the RESET# pulse. The device also resets the internal state machine to reading array data.

To ensure data integrity the operation that was interrupted should be reinitiated once the device is ready to accept another command sequence.

When RESET# is held at $V_{SS}$, the device draws CMOS standby current ($I_{CC4}$). If RESET# is held at $V_{IL}$, but not at $V_{SS}$, the standby current is greater.

RESET# may be tied to the system reset circuitry which enables the system to read the boot-up firmware from the Flash memory upon a system reset.

See Figure 11.5 on page 56 and Figure 11.8 on page 60 for timing diagrams.

## 7.8 Software Reset

Software reset is part of the command set (see Table 12.1 on page 66) that also returns the device to array read mode and must be used for the following conditions:

1. To exit Autoselect mode
2. To reset software when DQ5 goes high during write status operation that indicates program or erase cycle was not successfully completed
3. To exit sector lock/unlock operation.
4. To return to erase-suspend-read mode if the device was previously in Erase Suspend mode.
5. To reset software after any aborted operations

### Software Functions and Sample Code

**Table 7.19** Reset
(LLD Function = lld_ResetCmd)

| Cycle | Operation | Word Address | Data |
|---|---|---|---|
| Reset Command | Write | Base + xxxh | 00F0h |

***Note***
*Base = Base Address.*

The following is a C source code example of using the reset function. Refer to the *Spansion Low Level Driver User's Guide* (available on www.spansion.com) for general information on Spansion Flash memory software development guidelines.

```
/* Example: Reset (software reset of Flash state machine) */
  *( (UINT16 *)base_addr + 0x000 ) = 0x00F0;
```

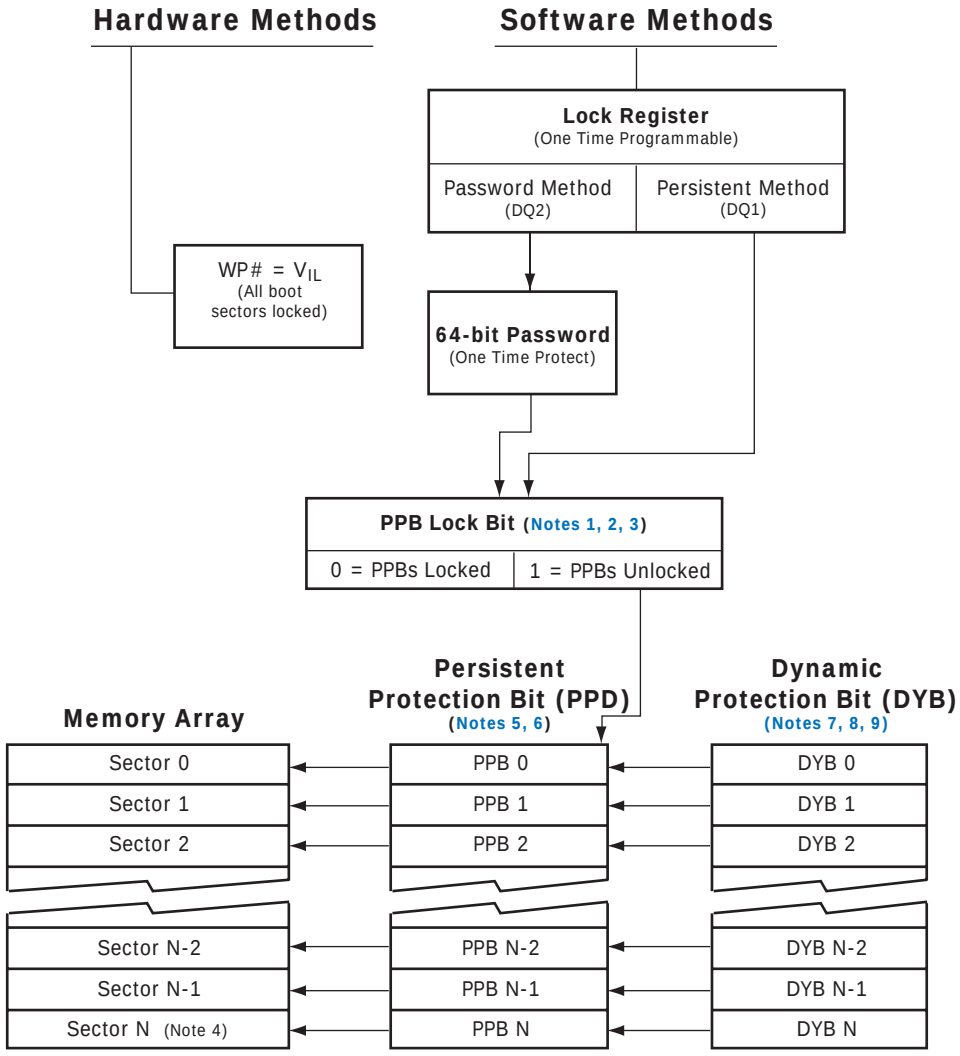
The following are additional points to consider when using the reset command:

- This command resets the banks to the read and address bits are ignored.
- Reset commands are ignored once erasure has begun until the operation is complete.
- Once programming begins, the device ignores reset commands until the operation is complete
- The reset command may be written between the cycles in a program command sequence before programming begins (prior to the third cycle). This resets the bank to which the system was writing to the read mode.
- If the program command sequence is written to a bank that is in the Erase Suspend mode, writing the reset command returns that bank to the erase-suspend-read mode.
- The reset command may be also written during an Autoselect command sequence.
- If a bank has entered the Autoselect mode while in the Erase Suspend mode, writing the reset command returns that bank to the erase-suspend-read mode.
- If DQ1 goes high during a Write Buffer Programming operation, the system must write the *Write to Buffer Abort Reset* command sequence to RESET the device to reading array data. The standard RESET command does not work during this condition.
- To exit the unlock bypass mode, the system must issue a two-cycle unlock bypass reset command sequence (see command tables for detail).

# 8. Advanced Sector Protection/Unprotection

The Advanced Sector Protection/Unprotection feature disables or enables programming or erase operations in any or all sectors and can be implemented through software and/or hardware methods, which are independent of each other. This section describes the various methods of protecting data stored in the memory array. An overview of these methods in shown in Figure 8.1 on page 44.

**Figure 8.1** Advanced Sector Protection/Unprotection




***Notes:***

1. *Bit is volatile, and defaults to 1 on reset.*
2. *Programming to 0 locks all PPBs to their current state.*
3. *Once programmed to 0, requires hardware reset to unlock.*
4. *N = Highest Address Sector.*
5. *0 = Sector Protected,*
   *1 = Sector Unprotected.*
6. *PPBs programmed individually, but cleared collectively.*
7. *0 = Sector Protected,*
   *1 = Sector Unprotected.*
8. *Protect effective only if PPB Lock Bit is unlocked and corresponding PPB is 1 (unprotected).*
9. *Volatile Bits: defaults to user choice upon power-up (see ordering options).*

## 8.1 Lock Register

As shipped from the factory, all devices default to the persistent mode when power is applied, and all sectors are unprotected, unless otherwise chosen through the DYB ordering option (see *Ordering Information* on page 9). The device programmer or host system must then choose which sector protection method to use. Programming (setting to *0*) any one of the following two one-time programmable, non-volatile bits locks the part permanently in that mode:

- Lock Register Persistent Protection Mode Lock Bit (DQ1)
- Lock Register Password Protection Mode Lock Bit (DQ2)

**Table 8.1** Lock Register

| Device | DQ15 – 05 | DQ4 | DQ3 | DQ2 | DQ1 | DQ0 |
|---|---|---|---|---|---|---|
| S29PL256N | Undefined | DYB Lock Boot Bit<br>0 = sectors power up protected<br>1 = sectors power up unprotected | PPB One-Time Programmable Bit<br>0 = All PPB erase command disabled<br>1 = All PPB Erase command enabled | Password Protection Mode Lock Bit | Persistent Protection Mode Lock Bit | Secured Silicon Sector Protection Bit |

For programming lock register bits see Table 12.2 on page 68.

### Notes

1. If the password mode is chosen, the password must be programmed before setting the corresponding lock register bit.
2. After the Lock Register Bits Command Set Entry command sequence is written, reads and writes for Bank A are disabled, while reads from other banks are allowed until exiting this mode.
3. If both lock bits are selected to be programmed (to zeros) at the same time, the operation aborts.
4. Once the Password Mode Lock Bit is programmed, the Persistent Mode Lock Bit is permanently disabled, and no changes to the protection scheme are allowed. Similarly, if the Persistent Mode Lock Bit is programmed, the Password Mode is permanently disabled.

After selecting a sector protection method, each sector can operate in any of the following three states:

1. *Constantly locked.* The selected sectors are protected and cannot be reprogrammed unless PPB lock bit is cleared via a password, hardware reset, or power cycle.
2. *Dynamically locked.* The selected sectors are protected and can be altered via software commands.
3. *Unlocked.* The sectors are unprotected and can be erased and/or programmed.

These states are controlled by the bit types described in Sections 8.2 – 8.6.

## 8.2 Persistent Protection Bits

The Persistent Protection Bits are unique and nonvolatile for each sector and have the same endurances as the Flash memory. Preprogramming and verification prior to erasure are handled by the device, and therefore do not require system monitoring.

**Notes**

1. Each PPB is individually programmed and all are erased in parallel.
2. Entry command disables reads and writes for the bank selected.
3. Reads within that bank return the PPB status for that sector.
4. Reads from other banks are allowed while writes are not allowed.
5. All Reads must be performed using the Asynchronous mode.
6. The specific sector addresses (A23 – A14 PL256N and A22 – A14 PL127N/PL129N) are written at the same time as the program command.
7. If the PPB Lock Bit is set, the PPB Program or erase command does not execute and times-out without programming or erasing the PPB.
8. There are no means for individually erasing a specific PPB and no specific sector address is required for this operation.
9. Exit command must be issued after the execution which resets the device to read mode and re-enables reads and writes for Bank A.
10. The programming state of the PPB for a given sector can be verified by writing a PPB Status Read Command to the device as described by the flow chart below.

## 8.3 Dynamic Protection Bits

Dynamic Protection Bits are volatile and unique for each sector and can be individually modified. DYBs only control the protection scheme for unprotected sectors that have their PPBs cleared (erased to *1*). By issuing the DYB Set or Clear command sequences, the DYBs are set (programmed to *0*) or cleared (erased to *1*), thus placing each sector in the protected or unprotected state respectively. This feature allows software to easily protect sectors against inadvertent changes yet does not prevent the easy removal of protection when changes are needed.

**Notes**

1. The DYBs can be set (programmed to *0*) or cleared (erased to *1*) as often as needed. When the parts are first shipped, the PPBs are cleared (erased to *1*) and upon power up or reset, the DYBs can be set or cleared depending upon the ordering option chosen.
2. If the option to clear the DYBs after power up is chosen, (erased to *1*), then the sectors may be modified depending upon the PPB state of that sector.
3. The sectors would be in the protected state If the option to set the DYBs after power up is chosen (programmed to *0*).
4. It is possible to have sectors that are persistently locked with sectors that are left in the dynamic state.
5. The DYB Set or Clear commands for the dynamic sectors signify protected or unprotected state of the sectors respectively. However, if there is a need to change the status of the persistently locked sectors, a few more steps are required. First, the PPB Lock Bit must be cleared by either putting the device through a power-cycle, or hardware reset. The PPBs can then be changed to reflect the desired settings. Setting the PPB Lock Bit once again locks the PPBs, and the device operates normally again.
6. To achieve the best protection, it is recommended to execute the PPB Lock Bit Set command early in the boot code and protect the boot code by holding WP# = $V_{IL}$. Note that the PPB and DYB bits have the same function when WP#/ACC = $V_{HH}$ as they do when WP#/ACC = $V_{IH}$.

## 8.4 Persistent Protection Bit Lock Bit

The Persistent Protection Bit Lock Bit is a global volatile bit for all sectors. When set (programmed to *0*), this bit locks all PPB and when cleared (programmed to *1*), unlocks each sector. There is only one PPB Lock Bit per device.

**Notes**

1. No software command sequence unlocks this bit unless the device is in the password protection mode; only a hardware reset or a power-up clears this bit.
2. The PPB Lock Bit must be set (programmed to *0*) only after all PPBs are configured to the desired settings.

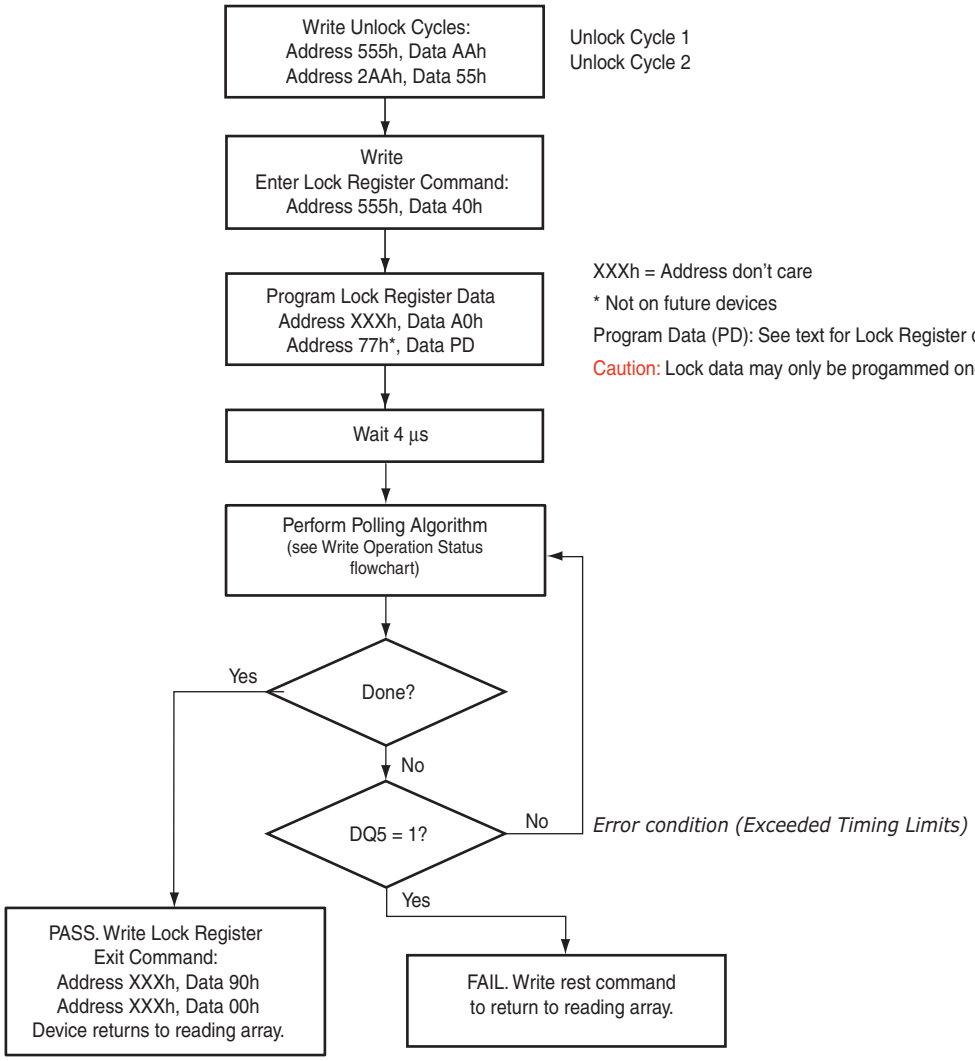
## 8.5 Password Protection Method

The Password Protection Method allows an even higher level of security than the Persistent Sector Protection Mode by requiring a 64-bit password for unlocking the device PPB Lock Bit. In addition to this password requirement, after power up and reset, the PPB Lock Bit is set *0* to maintain the password mode of operation. Successful execution of the Password Unlock command by entering the entire password clears the PPB Lock Bit, allowing for sector PPBs modifications.

**Notes**

1. There is no special addressing order required for programming the password. Once the Password is written and verified, the Password Mode Locking Bit must be set to prevent access.
2. The Password Program Command is only capable of programming *0*s. Programming a *1* after a cell is programmed as a *0* results in a time-out with the cell as a *0*.
3. The password is all *1*s when shipped from the factory.
4. All 64-bit password combinations are valid as a password.
5. There is no means to verify what the password is after it is set.
6. The Password Mode Lock Bit, once set, prevents reading the 64-bit password on the data bus and further password programming.
7. The Password Mode Lock Bit is not erasable.
8. The lower two address bits (A1 – A0) are valid during the Password Read, Password Program, and Password Unlock.
9. The exact password must be entered in order for the unlocking function to occur.
10. The Password Unlock command cannot be issued any faster than 1 µs at a time to prevent a hacker from running through all the 64-bit combinations in an attempt to correctly match a password.
11. Approximately 1 µs is required for unlocking the device after the valid 64-bit password is given to the device.
12. Password verification is only allowed during the password programming operation.
13. All further commands to the password region are disabled and all operations are ignored.
14. If the password is lost after setting the Password Mode Lock Bit, there is no way to clear the PPB Lock Bit.
15. Entry command sequence must be issued prior to any of any operation and it disables reads and writes for Bank A. Reads and writes for other banks excluding Bank A are allowed.
16. If the user attempts to program or erase a protected sector, the device ignores the command and returns to read mode.
17. A program or erase command to a protected sector enables status polling and returns to read mode without having modified the contents of the protected sector.
18. The programming of the DYB, PPB, and PPB Lock for a given sector can be verified by writing individual status read commands DYB Status, PPB Status, and PPB Lock Status to the device.

**Figure 8.2** Lock Register Program Algorithm




## 8.6 Advanced Sector Protection Software Examples

**Table 8.2** Sector Protection Schemes

| Unique Device PPB Lock Bit<br>0 = locked, 1 = unlocked | | Sector PPB<br>0 = protected<br>1 = unprotected | Sector DYB<br>0 = protected<br>1 = unprotected | Sector Protection Status |
|---|---|---|---|---|
| Any Sector | 0 | 0 | x | Protected through PPB |
| Any Sector | 0 | 0 | x | Protected through PPB |
| Any Sector | 0 | 1 | 1 | Unprotected |
| Any Sector | 0 | 1 | 0 | Protected through DYB |
| Any Sector | 1 | 0 | x | Protected through PPB |
| Any Sector | 1 | 0 | x | Protected through PPB |
| Any Sector | 1 | 1 | 0 | Protected through DYB |
| Any Sector | 1 | 1 | 1 | Unprotected |

Table 8.2 contains all possible combinations of the DYB, PPB, and PPB Lock Bit relating to the status of the sector. In summary, if the PPB Lock Bit is locked (set to *0*), no changes to the PPBs are allowed. The PPB Lock Bit can only be unlocked (reset to *1*) through a hardware reset or power cycle. See also Figure 8.1 on page 44 for an overview of the *Advanced Sector Protection* feature.

## 8.7 Hardware Data Protection Methods

The device offers data protection at the sector level via hardware control:

- When WP#/ACC is at $V_{IL}$, the four outermost sectors are locked (device specific).

There are additional methods by which intended or accidental erasure of any sectors can be prevented via hardware means. The following subsections describes these methods:

### 8.7.1 WP# Method

The Write Protect feature provides a hardware method of protecting the four outermost sectors. This function is provided by the WP#/ACC pin and overrides the previously discussed Sector Protection/Unprotection method.

If the system asserts $V_{IL}$ on the WP#/ACC pin, the device disables program and erase functions in the *outermost* boot sectors. The outermost boot sectors are the sectors containing both the lower and upper set of sectors in a dual-boot-configured device.

If the system asserts $V_{IH}$ on the WP#/ACC pin, the device reverts to whether the boot sectors were last set to be protected or unprotected. That is, sector protection or unprotection for these sectors depends on whether they were last protected or unprotected.

Note that the WP#/ACC pin must not be left floating or unconnected as inconsistent behavior of the device may result.

The WP#/ACC pin must be held stable during a command sequence execution

### 8.7.2 Low $V_{CC}$ Write Inhibit

When $V_{CC}$ is less than $V_{LKO}$, the device does not accept any write cycles. This protects data during $V_{CC}$ power-up and power-down.

The command register and all internal program/erase circuits are disabled, and the device resets to reading array data. Subsequent writes are ignored until $V_{CC}$ is greater than $V_{LKO}$. The system must provide the proper signals to the control inputs to prevent unintentional writes when $V_{CC}$ is greater than $V_{LKO}$.

### 8.7.3 Write Pulse *Glitch Protection*

Noise pulses of less than 3 ns (typical) on OE#, CE# or WE# do not initiate a write cycle.

### 8.7.4 Power-Up Write Inhibit

If WE# = CE# = RESET# = $V_{IL}$ and OE# = $V_{IH}$ during power up, the device does not accept commands on the rising edge of WE#. The internal state machine is automatically reset to the read mode on powerup.

# 9. Power Conservation Modes

## 9.1 Standby Mode

When the system is not reading or writing to the device, it can place the device in the standby mode. In this mode, current consumption is greatly reduced, and the outputs are placed in the high impedance state, independent of the OE# input. The device enters the CMOS standby mode when the CE# and RESET# inputs are both held at $V_{CC}$ ±0.2 V. The device requires standard access time ($t_{CE}$) for read access, before it is ready to read data. If the device is deselected during erasure or programming, the device draws active current until the operation is completed. $I_{CC3}$ in *DC Characteristics* on page 57 represents the standby current specification

## 9.2 Automatic Sleep Mode

The automatic sleep mode minimizes Flash device energy consumption while in asynchronous mode. the device automatically enables this mode when addresses remain stable for $t_{ACC}$ + 20 ns. The automatic sleep mode is independent of the CE#, WE#, and OE# control signals. Standard address access timings provide new data when addresses are changed. While in sleep mode, output data is latched and always available to the system. $I_{CC6}$ in *DC Characteristics* on page 57 represents the automatic sleep mode current specification.

## 9.3 Hardware RESET# Input Operation

The RESET# input provides a hardware method of resetting the device to reading array data. When RESET# is driven low for at least a period of $t_{RP}$, the device immediately terminates any operation in progress, tristates all outputs, resets the configuration register, and ignores all read/write commands for the duration of the RESET# pulse. The device also resets the internal state machine to reading array data. The operation that was interrupted should be reinitiated once the device is ready to accept another command sequence to ensure data integrity.

When RESET# is held at $V_{SS}$ ±0.2 V, the device draws CMOS standby current ($I_{CC4}$). If RESET# is held at $V_{IL}$ but not within $V_{SS}$ ±0.2 V, the standby current is greater.

RESET# may be tied to the system reset circuitry and thus, a system reset would also reset the Flash memory, enabling the system to read the boot-up firmware from the Flash memory.

## 9.4 Output Disable (OE#)

When the OE# input is at $V_{IH}$, output from the device is disabled. The outputs are placed in the high impedance state.

# 10. Secured Silicon Sector Flash Memory Region

The Secured Silicon Sector provides an extra Flash memory region that enables permanent part identification through an Electronic Serial Number (ESN). The Secured Silicon Sector is 256 words in length that consists of 128 words for factory data and 128 words for customer-secured areas. All Secured Silicon reads outside of the 256-word address range returns invalid data. The Factory Indicator Bit, DQ7, (at Autoselect address 03h) is used to indicate whether or not the Factory Secured Silicon Sector is locked when shipped from the factory. The Customer Indicator Bit (DQ6) is used to indicate whether or not the Customer Secured Silicon Sector is locked when shipped from the factory.

Note the following general conditions:

- While the Secured Silicon Sector access is enabled, simultaneous operations are allowed except for Bank A.
- On power up, or following a hardware reset, the device reverts to sending commands to the normal address space.
- Reads outside of sector 0 return memory array data.
- Sector 0 is remapped from the memory array to the Secured Silicon Sector array.
- Once the Secured Silicon Sector Entry Command is issued, the Secured Silicon Sector Exit command must be issued to exit Secured Silicon Sector Mode.
- The Secured Silicon Sector is not accessible when the device is executing an Embedded Program or Embedded Erase algorithm.

**Table 10.1** Secured Silicon Sector Addresses

| Sector | Sector Size | Address Range |
|---|---|---|
| Customer | 128 words | 000080h-0000FFh |
| Factory | 128 words | 000000h-00007Fh |

## 10.1 Factory Secured Silicon Sector

The Factory Secured Silicon Sector is always protected when shipped from the factory and has the Factory Indicator Bit (DQ7) permanently set to a *1*. This prevents cloning of a factory locked part and ensures the security of the ESN and customer code once the product is shipped to the field.

These devices are available pre programmed with one of the following:

- A random, 8-word secure ESN only within the Factory Secured Silicon Sector
- Customer code within the Customer Secured Silicon Sector through the Spansion™ programming service.
- Both a random, secure ESN and customer code through the Spansion programming service.

Customers may opt to have their code programmed through the Spansion programming services. Spansion programs the customer's code, with or without the random ESN. The devices are then shipped from the Spansion factory with the Factory Secured Silicon Sector and Customer Secured Silicon Sector permanently locked. Contact your local representative for details on using Spansion programming services.

## 10.2 Customer Secured Silicon Sector

The Customer Secured Silicon Sector is typically shipped unprotected (DQ6 set to *0*), allowing customers to utilize that sector in any manner they choose. If the security feature is not required, the Customer Secured Silicon Sector can be treated as an additional Flash memory space.

Please note the following:

- Once the Customer Secured Silicon Sector area is protected, the Customer Indicator Bit is permanently set to *1*.
- The Customer Secured Silicon Sector can be read any number of times, but can be programmed and locked only once. The Customer Secured Silicon Sector lock must be used with caution as once locked, there is no procedure available for unlocking the Customer Secured Silicon Sector area and none of the bits in the Customer Secured Silicon Sector memory space can be modified in any way.
- The accelerated programming (ACC) and unlock bypass functions are *not* available when programming the Customer Secured Silicon Sector, but are available when reading in Banks B through D.
- Once the Customer Secured Silicon Sector is locked and verified, the system must write the Exit Secured Silicon Sector Region command sequence which return the device to the memory array at sector 0.

## 10.3 Secured Silicon Sector Entry and Exit Command Sequences

The system can access the Secured Silicon Sector region by issuing the three-cycle Enter Secured Silicon Sector command sequence. The device continues to access the Secured Silicon Sector region until the system issues the four-cycle Exit Secured Silicon Sector command sequence.

See the Command Definition Tables: Table 12.1, *Memory Array Commands* on page 66, Table 12.2, *Sector Protection Commands* on page 68 for address and data requirements for both command sequences.

The Secured Silicon Sector Entry Command allows the following commands to be executed

- Read customer and factory Secured Silicon areas
- Program the customer Secured Silicon Sector

After the system has written the Enter Secured Silicon Sector command sequence, it may read the Secured Silicon Sector by using the addresses normally occupied by sector SA0 within the memory array. This mode of operation continues until the system issues the Exit Secured Silicon Sector command sequence, or until power is removed from the device.

## Software Functions and Sample Code

The following are C functions and source code examples of using the Secured Silicon Sector Entry, Program, and exit commands. Refer to the *Spansion Low Level Driver User Guide* (available soon on www.spansion.com) for general information on Spansion Flash memory software development guidelines.

**Table 10.2** Secured Silicon Sector Entry
(LLD Function = lld_SecSiSectorEntryCmd)

| Cycle | Operation | Word Address | Data |
|---|---|---|---|
| Unlock Cycle 1 | Write | Base + 555h | 00AAh |
| Unlock Cycle 2 | Write | Base + 2AAh | 0055h |
| Entry Cycle | Write | Base + 555h | 0088h |

***Note***
*Base = Base Address.*

```
/* Example: SecSi Sector Entry Command */
 *((UINT16 *)base_addr + 0x555) = 0x00AA;   /* write unlock cycle 1            */
 *((UINT16 *)base_addr + 0x2AA) = 0x0055;   /* write unlock cycle 2            */
 *((UINT16 *)base_addr + 0x555) = 0x0088;   /* write Secsi Sector Entry Cmd    */
```

**Table 10.3** Secured Silicon Sector Program
(LLD Function = lld_ProgramCmd)

| Cycle | Operation | Word Address | Data |
|---|---|---|---|
| Unlock Cycle 1 | Write | Base + 555h | 00AAh |
| Unlock Cycle 2 | Write | Base + 2AAh | 0055h |
| Program Setup | Write | Base + 555h | 00A0h |
| Program | Write | Word Address | Data Word |

***Note***
*Base = Base Address.*

```
/* Once in the SecSi Sector mode, you program */
/* words using the programming algorithm.     */
```

**Table 10.4** Secured Silicon Sector Exit
(LLD Function = lld_SecSiSectorExitCmd)

| Cycle | Operation | Word Address | Data |
|---|---|---|---|
| Unlock Cycle 1 | Write | Base + 555h | 00AAh |
| Unlock Cycle 2 | Write | Base + 2AAh | 0055h |
| Exit Cycle | Write | Base + 555h | 0090h |

***Note***
*Base = Base Address.*

```
/* Example: SecSi Sector Exit Command */
 *((UINT16 *)base_addr + 0x555) = 0x00AA;   /* write unlock cycle 1            */
 *((UINT16 *)base_addr + 0x2AA) = 0x0055;   /* write unlock cycle 2            */
 *((UINT16 *)base_addr + 0x555) = 0x0090;   /* write SecSi Sector Exit cycle 3 */
 *((UINT16 *)base_addr + 0x000) = 0x0000;   /* write SecSi Sector Exit cycle 4 */
```

# 11. Electrical Specifications

## 11.1 Absolute Maximum Ratings

| | |
|---|---|
| Storage Temperature, Plastic Packages | –65°C to +150°C |
| Ambient Temperature with Power Applied | –65°C to +125°C |
| Voltage with Respect to Ground:<br>All Inputs and I/Os except as noted below (Note 1) | –0.5 V to $V_{IO}$ + 0.5 V |
| $V_{CC}$ (Note 1) | –0.5 V to +4.0 V |
| $V_{IO}$ (Note 1) | –0.5 V to +4.0V |
| ACC (Note 2) | –0.5 V to +10.5 V |
| Output Short Circuit Current (Note 3) | 200 mA |

***Notes***

1. *Minimum DC voltage on input or I/Os is –0.5 V. During voltage transitions, inputs or I/Os may undershoot $V_{SS}$ to –2.0 V for periods of up to 20 ns. See Figure 11.1 on page 54. Maximum DC voltage on input or I/Os is $V_{CC}$ + 0.5 V. During voltage transitions outputs may overshoot to $V_{CC}$ + 2.0 V for periods up to 20 ns. See Figure 11.2 on page 54.*
2. *Minimum DC input voltage on pin WP#/ACC is –0.5 V. During voltage transitions, WP#/ACC may overshoot $V_{SS}$ to 2.0 V for periods of up to 20 ns. See Figure 11.1 on page 54. Maximum DC voltage on pin WP#/ACC is +9.5 V, which may overshoot to 10.5 V for periods up to 20 ns.*
3. *No more than one output may be shorted to ground at a time. Duration of the short circuit should not be greater than one second.*
4. *Stresses above those listed under Absolute Maximum Ratings on page 54 may cause permanent damage to the device. This is a stress rating only; functional operation of the device at these or any other conditions above those indicated in the operational sections of this data sheet is not implied. Exposure of the device to absolute maximum rating conditions for extended periods may affect device reliability.*

**Figure 11.1** Maximum Negative Overshoot Waveform

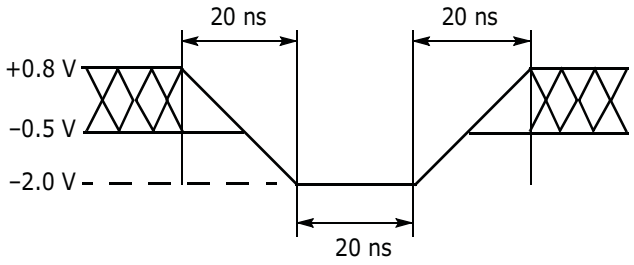
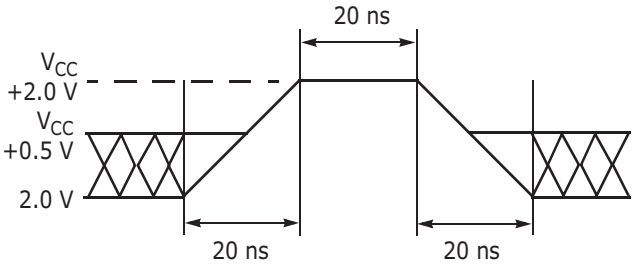



**Figure 11.2** Maximum Positive Overshoot Waveform



## 11.2 Operating Ranges

| | |
|---|---|
| Wireless (W) Devices | –25°C to +85°C |
| Ambient Temperature ($T_A$) | |
| Industrial (I) Devices | –40°C to +85°C |
| Ambient Temperature ($T_A$) | |
| $V_{CC}$ Supply Voltages (Note 3) | +2.7 V to 3.1 V or |
| | +2.7 V to +3.6 V |

***Notes***

1. *Operating ranges define those limits between which the functionality of the device is guaranteed.*
2. *For all AC and DC specifications, $V_{IO} = V_{CC}$.*
3. *Voltage range of 2.7 – 3.1 V valid for PL-N MCP products.*

## 11.3 Test Conditions

**Figure 11.3** Test Setup

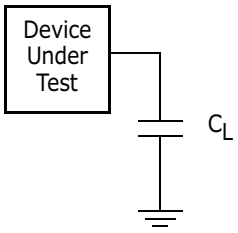



**Table 11.1** Test Specifications

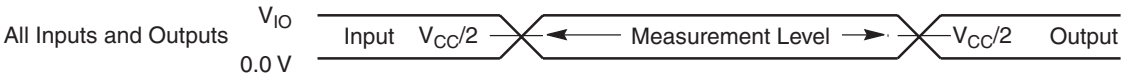
| Test Condition | | All Speeds | Unit |
|---|---|---|---|
| Output Load Capacitance, $C_L$ (including jig capacitance) | | 30 | pF |
| Input Rise and Fall Times | $V_{CC}$ = 3.0 V | 5 | ns |
| Input Pulse Levels | $V_{CC}$ = 3.0 V | 0.0 – 3.0 | V |
| Input timing measurement reference levels | | $V_{CC}/2$ | V |
| Output timing measurement reference levels | | $V_{CC}/2$ | V |

## 11.4 Key to Switching Waveforms

| Waveform | Inputs | Outputs |
|---|---|---|
| | Steady | |
| | Changing from H to L | |
| | Changing from L to H | |
| | Don't Care, Any Change Permitted | Changing, State Unknown |
| | Does Not Apply | Center Line is High Impedance State (High Z) |

## 11.5 Switching Waveforms

**Figure 11.4** Input Waveforms and Measurement Levels



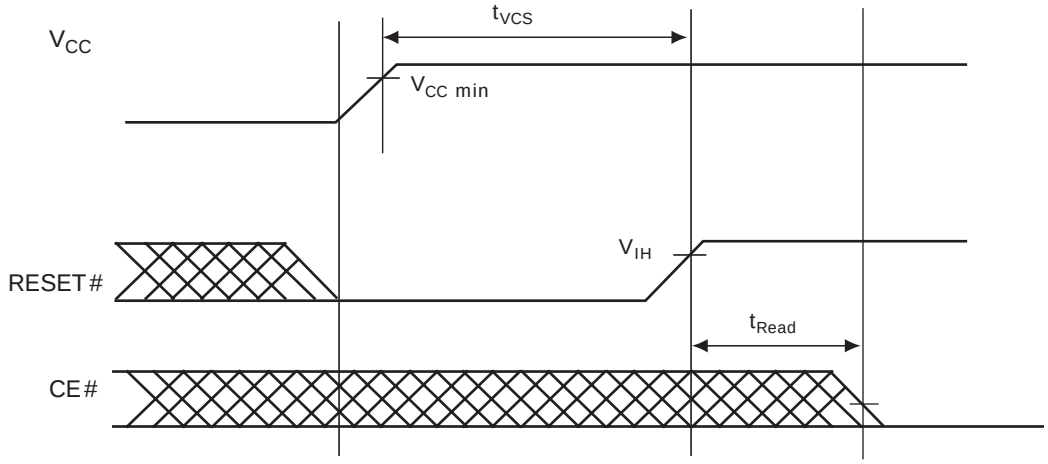

## 11.6 $V_{CC}$ Power Up

| Parameter | Description | Test Setup | Speed | Unit |
|---|---|---|---|---|
| $t_{VCS}$ | $V_{CC}$ Setup Time | Min | 250 | µs |
| $t_{READ}$ | Time between RESET# high and CE# low | Min | 200 | ns |

***Notes***

1. *$V_{CC}$ ramp rate must exceed 1 V/400 µs.*
2. *$V_{IO}$ is internally connected to $V_{CC}$.*

**Figure 11.5** $V_{CC}$ Power-Up Diagram



# 11.7 DC Characteristics

## 11.7.1 DC Characteristics ($V_{CC}$ = 2.7 V to 3.6 V)

### (CMOS Compatible)

| Parameter Symbol | Parameter Description (Notes) | Test Conditions | | Min (Note 2) | Typ (Note 2) | Max | Unit |
|---|---|---|---|---|---|---|---|
| $I_{LI}$ | Input Load Current | $V_{IN} = V_{SS}$ to $V_{CC}$, $V_{CC} = V_{CC\ max}$ (6) | | | | ±2.0 | µA |
| $I_{LO}$ | Output Leakage Current | $V_{OUT} = V_{SS}$ to $V_{CC}$, OE# = $V_{IH}$<br>$V_{CC} = V_{CC\ max}$ (6) | | | | ±1.0 | µA |
| $I_{CC1}$ | $V_{CC}$ Active Read Current (1, 3) | OE# = $V_{IH}$,<br>$V_{CC} = V_{CC\ max}$ (1, 6) | 5 MHz | | 30 | 45 | mA |
| $I_{CC2}$ | $V_{CC}$ Active Write Current (3) | OE# = $V_{IH}$, WE# = $V_{IL}$ | | | 25 | 50 | mA |
| $I_{CC3}$ | $V_{CC}$ Standby Current | CE# (7), RESET#,<br>WP#/ACC = $V_{CC} \pm 0.3$ V | | | 20 | 40 | µA |
| $I_{CC4}$ | $V_{CC}$ Reset Current | RESET# = $V_{SS} \pm 0.3$ V | | | 300 | 500 | µA |
| $I_{CC5}$ | Automatic Sleep Mode (4) | $V_{IH} = V_{CC} \pm 0.3$ V; $V_{IL} = V_{SS} \pm 0.3$ V | | | 20 | 40 | µA |
| $I_{CC6}$ | $V_{CC}$ Active Read-While-Write Current (1) | OE# = $V_{IH}$ | 5 MHz | | 35 | 50 | mA |
| $I_{CC7}$ | $V_{CC}$ Active Program-While-Erase-Suspended Current (5) | OE# = $V_{IH}$ | | | 27 | 55 | mA |
| $I_{CC8}$ | $V_{CC}$ Active Page Read Current | OE# = $V_{IH}$, 8 word Page Read | 40 MHz | | 6 | 10 | mA |
| $V_{IL}$ | Input Low Voltage | $V_{CC}$ = 2.7 to 3.6 V | | –0.5 | | 0.8 | V |
| $V_{IH}$ | Input High Voltage | $V_{CC}$ = 2.7 to 3.6 V | | 2.0 | | $V_{CC}$ + 0.3 | V |
| $V_{HH}$ | Voltage for ACC Program Acceleration | $V_{CC}$ = 3.0 V ±10% (6) | | 8.5 | | 9.5 | V |
| $V_{OL}$ | Output Low Voltage | $I_{OL}$ = 100 µA, $V_{CC} = V_{CC\ min}$ (6) | | | | 0.1 | V |
| $V_{OH}$ | Output High Voltage | $I_{OH}$ = –100 µA (6) | | $V_{CC}$ – 0.2 | | | V |
| $V_{LKO}$ | Low $V_{CC}$ Lock-Out Voltage (5) | | | 2.3 | | 2.5 | V |

***Notes***

1. *The $I_{CC}$ current listed is typically less than 5 mA/MHz, with OE# at $V_{IH}$.*
2. *Maximum $I_{CC}$ specifications are tested with $V_{CC} = V_{CC}$ max, $T_A = T_A$max. Typical $I_{CC}$ specifications are with typical $V_{CC}$=3.0 V, $T_A$ = +25°C.*
3. *$I_{CC}$ is active while Embedded Erase or Embedded Program is in progress.*
4. *Automatic sleep mode enables the low power mode when addresses remain stable for $t_{ACC}$ +30 ns. Typical sleep mode current is 1 µA.*
5. *Not 100% tested.*
6. *The data in the table is for $V_{CC}$ range 2.7 V to 3.6 V (recommended for standalone applications).*
7. *CE1# and CE2# for the PL129N.*

## 11.7.2 DC Characteristics ($V_{CC}$ = 2.7 V to 3.1 V)

### (CMOS Compatible)

| Parameter Symbol | Parameter Description (Notes) | Test Conditions | | Min | Typ | Max | Unit |
|---|---|---|---|---|---|---|---|
| $I_{LI}$ | Input Load Current | $V_{IN} = V_{SS}$ to $V_{CC}$, $V_{CC} = V_{CC\ max}$ (6) | | | | ±2 | µA |
| $I_{LO}$ | Output Leakage Current | $V_{OUT} = V_{SS}$ to $V_{CC}$, OE# = $V_{IH}$<br>$V_{CC} = V_{CC\ max}$ (6) | | | | ±1 | µA |
| $I_{CC1}$ | $V_{CC}$ Active Read Current (1, 2) | OE# = $V_{IH}$,<br>$V_{CC} = V_{CC\ max}$ (1, 6) | 5 MHz | | 28 | 40 | mA |
| $I_{CC2}$ | $V_{CC}$ Active Write Current (2, 3) | OE# = $V_{IH}$, WE# = $V_{IL}$ | | | 22 | 40 | mA |
| $I_{CC3}$ | $V_{CC}$ Standby Current (2) | CE# (7), RESET#, WP#/ACC<br>= $V_{CC}$ ± 0.3 V | | | 20 | 40 | µA |
| $I_{CC4}$ | $V_{CC}$ Reset Current (2) | RESET# = $V_{SS}$ ± 0.3 V | | | 300 | 500 | µA |
| $I_{CC5}$ | Automatic Sleep Mode (2, 4) | $V_{IH} = V_{CC}$ ±0.3 V; $V_{IL} = V_{SS}$ ± 0.1 V | | | 20 | 40 | µA |
| $I_{CC6}$ | $V_{CC}$ Active Read-While-Write Current (1, 2) | OE# = $V_{IH}$ | 5 MHz | | 33 | 45 | mA |
| $I_{CC7}$ | $V_{CC}$ Active Program-While-Erase-Suspended Current (2, 5) | OE# = $V_{IH}$ | | | 24 | 45 | mA |
| $I_{CC8}$ | $V_{CC}$ Active Page Read Current (2) | OE# = $V_{IH}$,<br>8 word Page Read | 40 MHz | | 6 | 9 | mA |
| $V_{IL}$ | Input Low Voltage | $V_{CC}$ = 2.7 to 3.6 V | | –0.5 | | 0.8 | V |
| $V_{IH}$ | Input High Voltage | $V_{CC}$ = 2.7 to 3.6 V | | 2.0 | | $V_{CC}$ + 0.3 | V |
| $V_{HH}$ | Voltage for ACC Program Acceleration | $V_{CC}$ = 3.0 V ±10% (6) | | 8.5 | | 9.5 | V |
| $V_{OL}$ | Output Low Voltage | $I_{OL}$ = 100 µA, $V_{CC} = V_{CC\ min}$ (6) | | | | 0.1 | V |
| $V_{OH}$ | Output High Voltage | $I_{OH}$ = –100 µA (6) | | $V_{CC}$ – 0.2 | | | V |
| $V_{LKO}$ | Low $V_{CC}$ Lock-Out Voltage (5) | | | 2.3 | | 2.5 | V |

***Notes***

1. *The $I_{CC}$ current listed is typically less than 5 mA/MHz, with OE# at $V_{IH}$.*
2. *Maximum $I_{CC}$ specifications are tested with $V_{CC} = V_{CC}$ max, $T_A = T_A$max. Typical $I_{CC}$ specifications are with typical $V_{CC}$=2.9 V, $T_A$ = +25°C.*
3. *$I_{CC}$ active while Embedded Erase or Embedded Program is in progress.*
4. *Automatic sleep mode enables the low power mode when addresses remain stable for $t_{ACC}$ + 30 ns. Typical sleep mode current is 1 µA.*
5. *Not 100% tested.*
6. *Data in table is for $V_{CC}$ range 2.7 V to 3.1 V (recommended for MCP applications)*
7. *CE1# and CE2# for the PL129N.*

## 11.8 AC Characteristics

### 11.8.1 Read Operations

| Parameter | | Description (Notes) | | Test Setup | | Speed Options | | | Unit |
|---|---|---|---|---|---|---|---|---|---|
| JEDEC | Std. | | | | | 65 | 70 | 80 | |
| $t_{AVAV}$ | $t_{RC}$ | Read Cycle Time (1) | | | Min | 65 | 70 | 80 | ns |
| $t_{AVQV}$ | $t_{ACC}$ | Address to Output Delay | | CE#, OE# = $V_{IL}$ | Max | 65 | 70 | 80 | ns |
| $t_{ELQV}$ | $t_{CE}$ | Chip Enable to Output Delay (5) | | OE# = $V_{IL}$ | Max | 65 | 70 | 80 | ns |
| | $t_{PACC}$ | Page Access Time | | | Max | 25 | 30 | 30 | ns |
| $t_{GLQV}$ | $t_{OE}$ | Output Enable to Output Delay | | | Max | 25 | 30 | 30 | ns |
| $t_{EHQZ}$ | $t_{DF}$ | Chip Enable to Output High Z (3) | | | Max | 16 | | | ns |
| $t_{GHQZ}$ | $t_{DF}$ | Output Enable to Output High Z (1, 3) | | | Max | 16 | | | ns |
| $t_{AXQX}$ | $t_{OH}$ | Output Hold Time From Addresses, CE# or OE#, Whichever Occurs First (3) | | | Min | 5 | | | ns |
| | $t_{OEH}$ | Output Enable Hold Time (1) | Read | | Min | 0 | | | ns |
| | | | Toggle and Data# Polling | | Min | 10 | | | ns |

***Notes***

1. *Not 100% tested.*
2. *See Figure 11.3 on page 55 and Table 11.1 on page 55 for test specifications*
3. *Measurements performed by placing a 50 ohm termination on the data pin with a bias of $V_{CC}$ /2. The time from OE# high to the data bus driven to $V_{CC}$ /2 is taken as $t_{DF}$.*
4. *For 70pf Output Load Capacitance, 2 ns is added to the above $t_{ACC}$ ,$t_{CE}$ ,$t_{PACC}$ ,$t_{OE}$ values for all speed grades*
5. *CE1# and CE2# for the PL129N.*

### 11.8.2 Read Operation Timing Diagrams

**Figure 11.6** Read Operation Timings

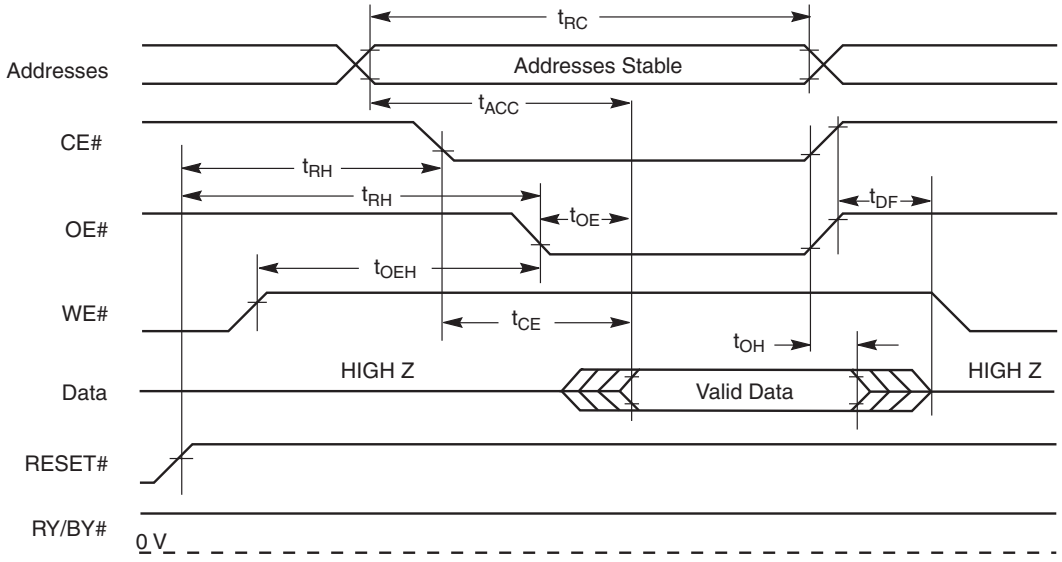
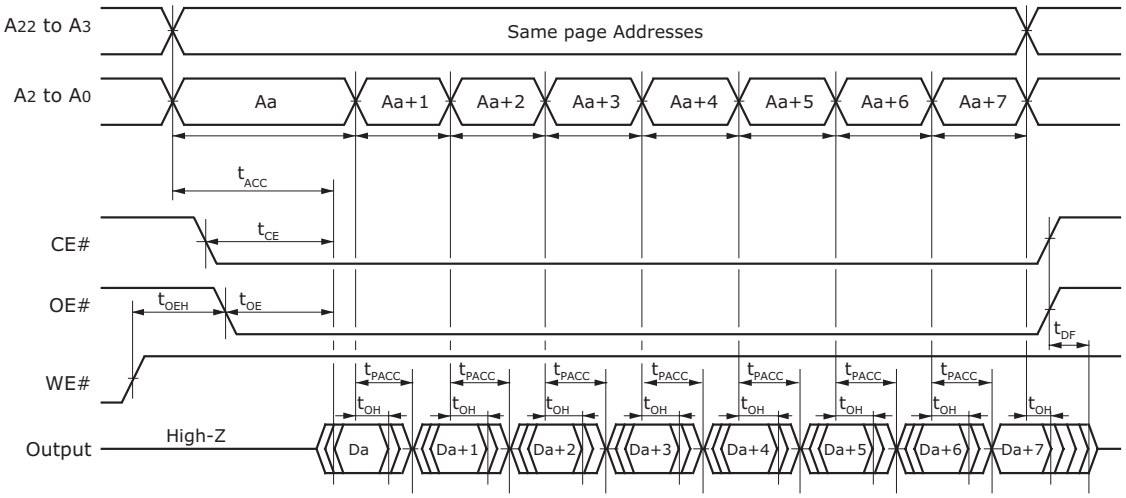


**Figure 11.7** Page Read Operation Timings

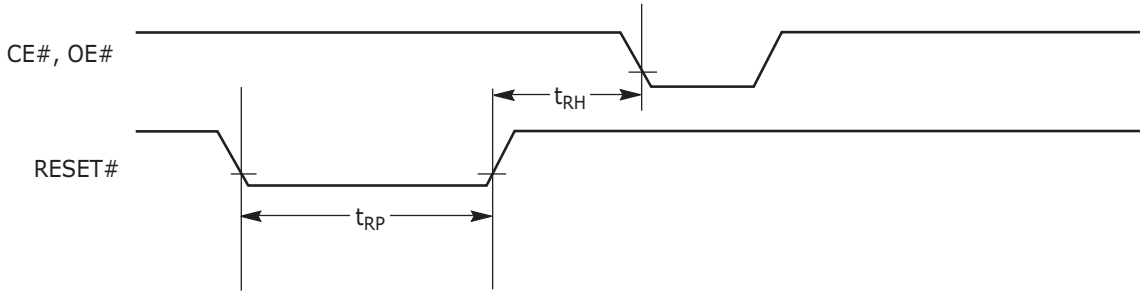



## 11.8.3 Hardware Reset (RESET#)

| Parameter | | Description | | All Speed Options | Unit |
|---|---|---|---|---|---|
| JEDEC | Std. | | | | |
| | $t_{RP}$ | RESET# Pulse Width | Min | 30 | µs |
| | $t_{RH}$ | Reset High Time Before Read (See Note) | Min | 200 | ns |

***Note***
*Not 100% tested.*

**Figure 11.8** Reset Timings



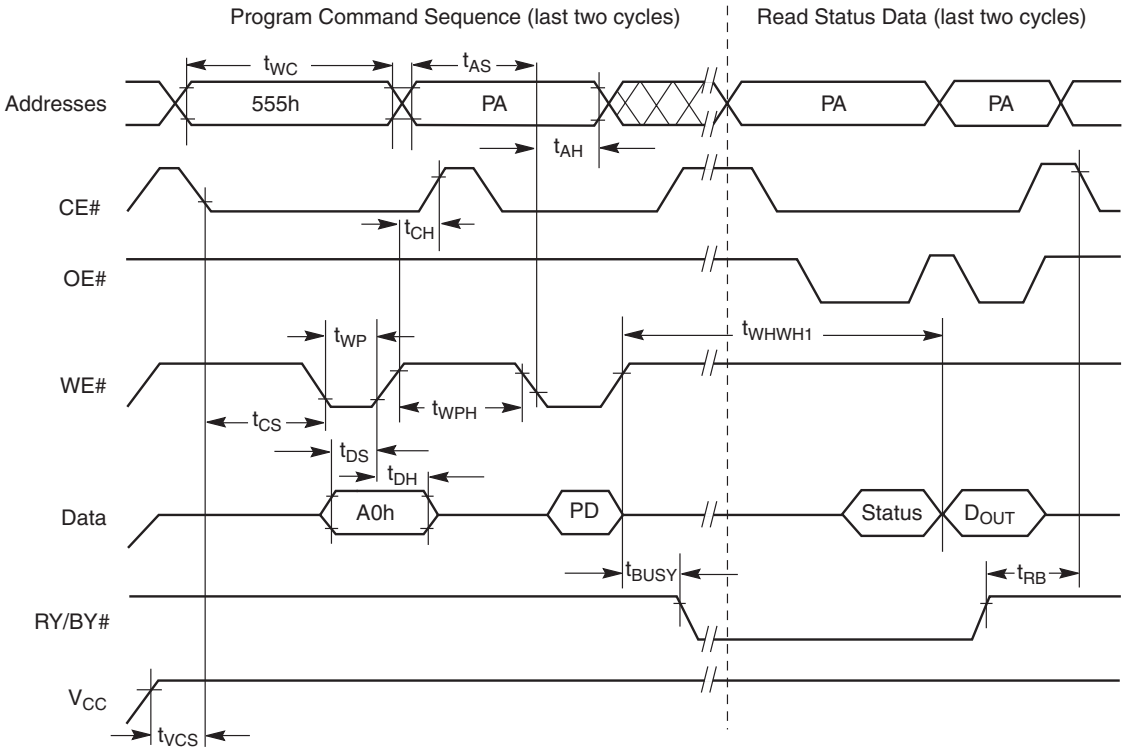
## 11.8.4 Erase/Program Timing

| Parameter | | Description (Notes) | | Speed Options | | | Unit |
|---|---|---|---|---|---|---|---|
| JEDEC | Std | | | 65 | 70 | 80 | |
| $t_{AVAV}$ | $t_{WC}$ | Write Cycle Time (1) | Min | 65 | 70 | 80 | ns |
| $t_{AVWL}$ | $t_{AS}$ | Address Setup Time | Min | 0 | | | ns |
| | $t_{ASO}$ | Address Setup Time to OE# low during toggle bit polling | Min | 15 | | | ns |
| $t_{WLAX}$ | $t_{AH}$ | Address Hold Time | Min | 35 | | | ns |
| | $t_{AHT}$ | Address Hold Time From CE# or OE# high during toggle bit polling | Min | 0 | | | ns |
| $t_{DVWH}$ | $t_{DS}$ | Data Setup Time | Min | 30 | | | ns |
| $t_{WHDX}$ | $t_{DH}$ | Data Hold Time | Min | 0 | | | ns |
| | $t_{OEPH}$ | Output Enable High during toggle bit polling | Min | 10 | | | ns |
| $t_{GHWL}$ | $t_{GHWL}$ | Read Recovery Time Before Write (OE# High to WE# Low) | Min | 0 | | | ns |
| $t_{ELWL}$ | $t_{CS}$ | CE# Setup Time | Min | 0 | | | ns |
| $t_{WHEH}$ | $t_{CH}$ | CE# Hold Time | Min | 0 | | | ns |
| $t_{WLWH}$ | $t_{WP}$ | Write Pulse Width | Min | 40 | | | ns |
| $t_{WHDL}$ | $t_{WPH}$ | Write Pulse Width High | Min | 25 | | | ns |
| | $t_{SR/W}$ | Latency Between Read and Write Operations | Min | 0 | | | ns |
| $t_{WHWH1}$ | $t_{WHWH1}$ | Programming Operation | Typ | 40 | | | µs |
| $t_{WHWH1}$ | $t_{WHWH1}$ | Accelerated Programming Operation | Typ | 24 | | | µs |
| $t_{WHWH2}$ | $t_{WHWH2}$ | Sector Erase Operation | Typ | 1.6 | | | sec |
| | $t_{VHH}$ | $V_{HH}$ Rise and Fall Times | Min | 250 | | | ns |
| | $t_{RB}$ | Write Recovery Time from RY/BY# | Min | 0 | | | ns |
| | $t_{BUSY}$ | Program/Erase Valid to RY/BY# Delay | Max | 90 | | | ns |
| | $t_{WEP}$ | Noise Pulse Margin on WE# | Max | 3 | | | ns |
| | $t_{SEA}$ | Sector Erase Accept Time-out | Max | 50 | | | µs |
| | $t_{ESL}$ | Erase Suspend Latency | Max | 20 | | | µs |
| | $t_{PSL}$ | Program Suspend Latency | Max | 20 | | | µs |
| | $t_{ASP}$ | Toggle Time During Sector Protection | Typ | 100 | | | µs |
| | $t_{PSP}$ | Toggle Time During Programming Within a Protected Sector | Typ | 1 | | | µs |

***Notes***

1. *Not 100% tested.*
2. *In program operation timing, addresses are latched on the falling edge of WE#.*
3. *See Program/Erase Operations on page 25 for more information.*
4. *Does not include the preprogramming time.*

**Figure 11.9** Program Operation Timings




***Note***
*PA = program address, PD = program data, $D_{OUT}$ is the true data at the program address*

**Figure 11.10** Accelerated Program Timing Diagram

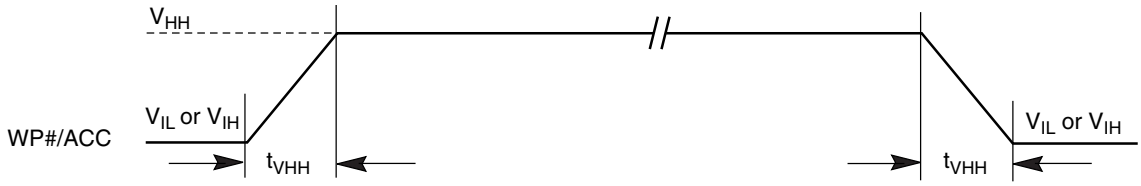
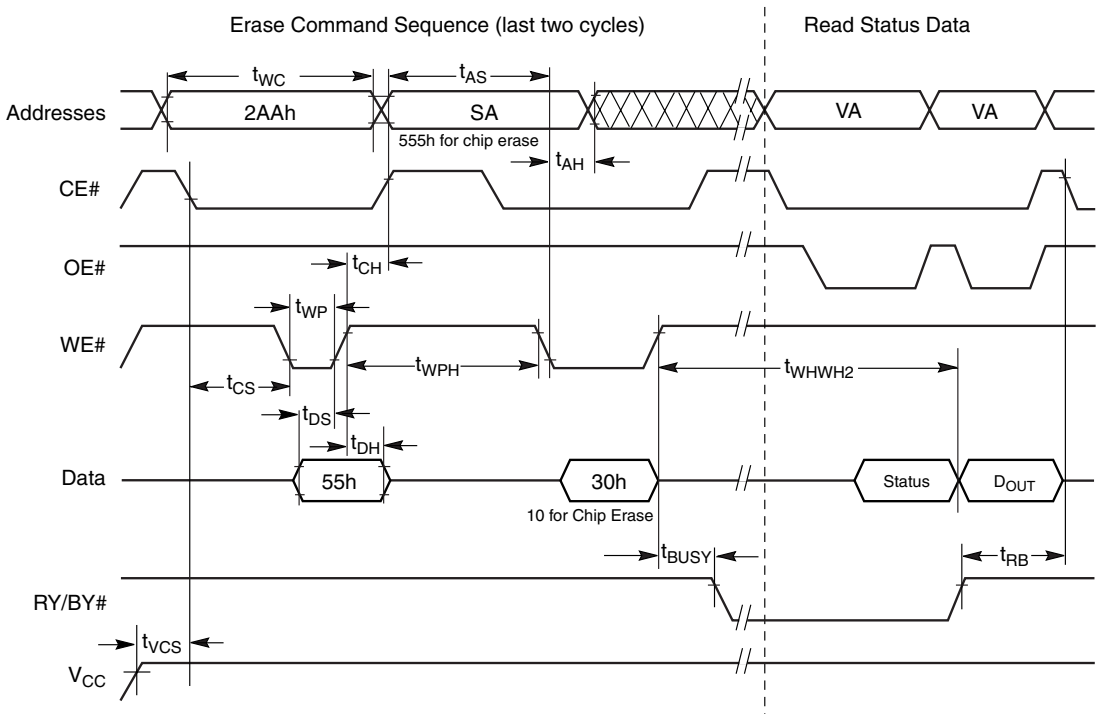


**Figure 11.11** Chip/Sector Erase Operation Timings




***Note***

*SA = sector address (for Sector Erase), VA = Valid Address for reading status data (see Write Operation Status on page 37)*

**Figure 11.12** Back-to-back Read/Write Cycle Timings

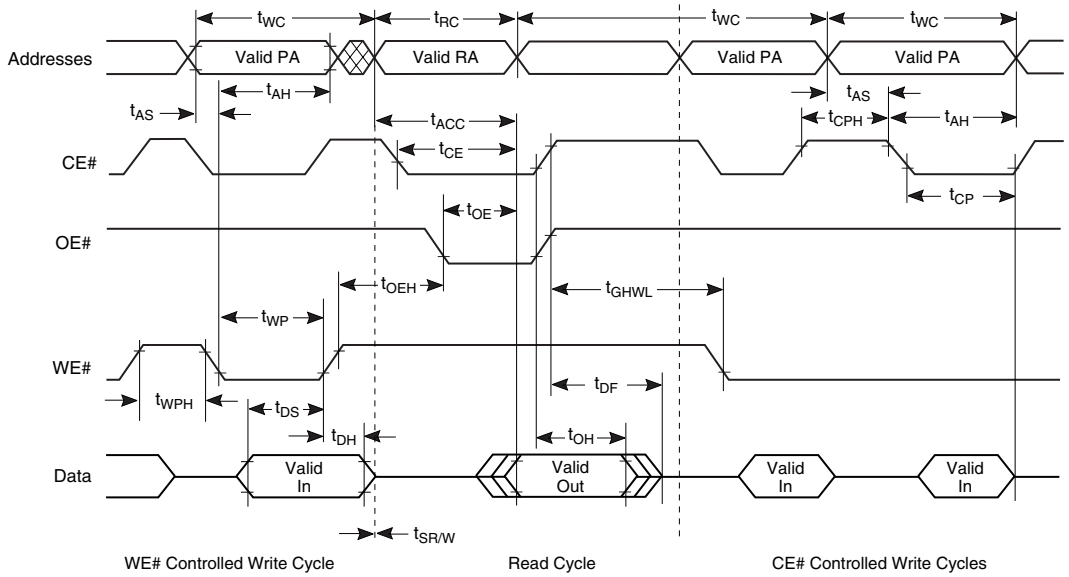
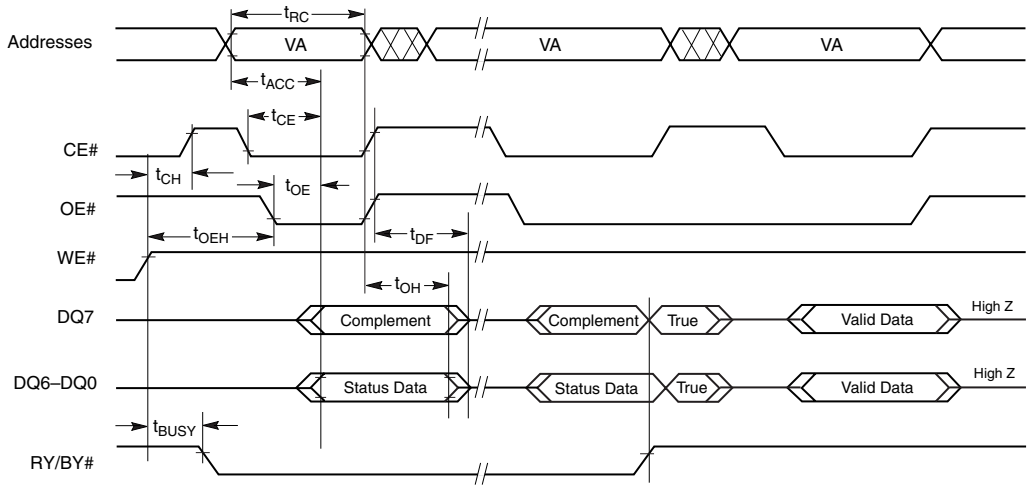
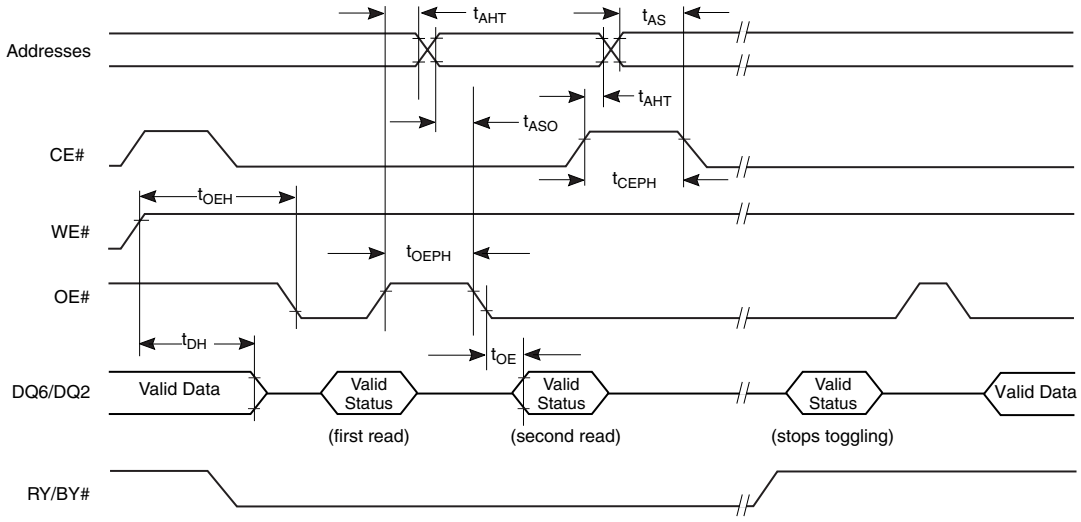


**Figure 11.13** Data# Polling Timings (During Embedded Algorithms)



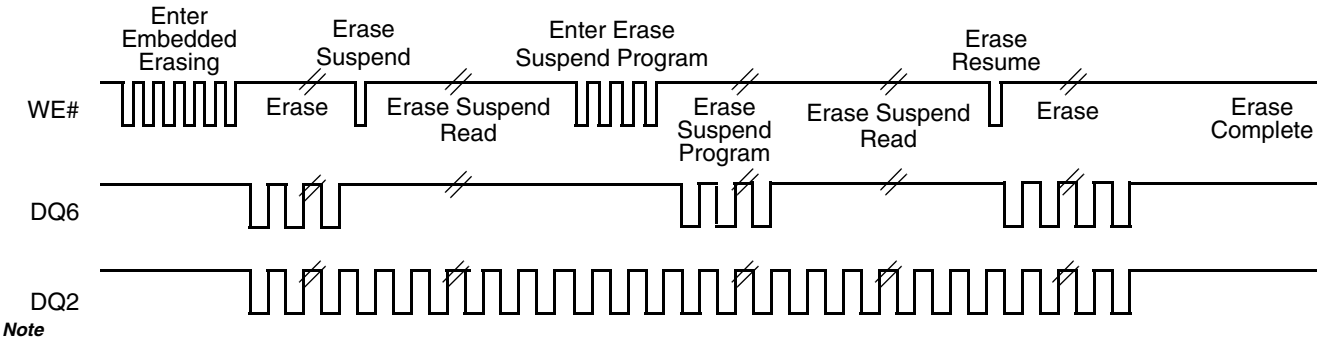

***Note***
*VA = Valid address. Illustration shows first status cycle after command sequence, last status read cycle, and array data read cycle*

**Figure 11.14** Toggle Bit Timings (During Embedded Algorithms)




***Note***
*VA = Valid address; not required for DQ6. Illustration shows first two status cycle after command sequence, last status read cycle, and array data read cycle*

**Figure 11.15** DQ2 vs. DQ6




***Note***
*DQ2 toggles only when read at an address within an erase-suspended sector. The system may use OE# or CE# to toggle DQ2 and DQ6.*

## 11.8.5 Erase and Programming Performance

| Parameter (Notes) | | Device Condition | Typ (Note 1) | Max (Note 2) | Unit | Comments (Notes) |
|---|---|---|---|---|---|---|
| Sector Erase Time | 128 Kword | $V_{CC}$ | 1.6 | 7 | s | Excludes 00h programming prior to erasure (4) |
| | | ACC | 1.6 | 7 | | |
| | 32 Kword | $V_{CC}$ | 0.3 | 4 | | |
| | | ACC | 0.3 | 4 | | |
| Chip Erase Time | | $V_{CC}$ | 202 (PL256N)<br>100 (PL127N)<br>100(PL129N) | 900 (PL256N)<br>450 (PL127N)<br>450 (PL129N) | s | |
| | | ACC | 130 (PL256N)<br>65 (PL127N)<br>65 (PL129N) | 512 (PL256N)<br>256 (PL127N)<br>256 (PL129N) | | |
| Word Programming Time | | $V_{CC}$ | 40 | 400 | µs | Excludes system level overhead (5) |
| | | ACC | 24 | 240 | | |
| Effective Word Programming Time utilizing Program Write Buffer | | $V_{CC}$ | 9.4 | 94 | µs | |
| | | ACC | 6 | 60 | | |
| Total 32-Word Buffer Programming Time | | $V_{CC}$ | 300 | 3000 | µs | |
| | | ACC | 192 | 1920 | | |
| Chip Programming Time using 32-Word Buffer (3) | | $V_{CC}$ | 157.3 (PL256N)<br>78.6 (PL127N)<br>78.6 (PL129N) | 315 (PL256N)<br>158 (PL127N)<br>158 (PL129N) | s | Excludes system level overhead (5) |
| | | ACC | 100 (PL256N)<br>50 (PL127N)<br>50 (PL129N) | 200 (PL256N)<br>100 (PL127N)<br>100 (PL129N) | | |
| Erase Suspend/Erase Resume | | | | <20 | µs | |
| Program Suspend/Program Resume | | | | <20 | µs | |

***Notes***

1. *Typical program and erase times assume the following conditions: 25°C, 3.0 V $V_{CC}$, 10,000 cycles. Additionally, programming typicals assume checkerboard pattern. All values are subject to change.*
2. *Under worst case conditions of 90°C, $V_{CC}$ = 2.7 V, 100,000 cycles. All values are subject to change.*
3. *The typical chip programming time is considerably less than the maximum chip programming time listed, since most bytes program faster than the maximum program times listed.*
4. *In the pre-programming step of the Embedded Erase algorithm, all bytes are programmed to 00h before erasure.*
5. *System-level overhead is the time required to execute the two- or four-bus-cycle sequence for the program command. See Table 12.1 on page 66 and Table 12.2 on page 68 for further information on command definitions.*
6. *Contact the local sales office for minimum cycling endurance values in specific applications and operating conditions.*
7. *See Application Note Erase Suspend/Resume Timing for more details.*
8. *Word programming specification is based upon a single word programming operation not utilizing the write buffer.*

## 11.8.6 BGA Ball Capacitance

| Parameter | Parameter Description | Test Setup | Typ | Max | Unit |
|---|---|---|---|---|---|
| $C_{IN}$ | Input Capacitance | $V_{IN}$ = 0 | 7 | 10 | pF |
| $C_{OUT}$ | Output Capacitance | $V_{OUT}$ = 0 | 8 | 12 | pF |
| $C_{IN2}$ | Control Pin Capacitance | $V_{IN}$ = 0 | 8 | 11 | pF |

***Notes***

1. *Sampled, not 100% tested.*
2. *Test conditions $T_A$ = 25°C, f = 1.0 MHz.*

# 12. Appendix

This section contains information relating to software control or interfacing with the Flash device. For additional information and assistance regarding software, see *Additional Resources* on page 17, or explore the Web at www.spansion.com.

**Table 12.1** Memory Array Commands

| Command Sequence (Notes) | | Cycles | Bus Cycles (Notes 1 – 6) | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | First | | Second | | Third | | Fourth | | Fifth | | Sixth | |
| | | | Addr | Data | Addr | Data | Addr | Data | Addr | Data | Addr | Data | Addr | Data |
| Read (7) | | 1 | RA | RD | | | | | | | | | | |
| Reset (8) | | 1 | XXX | F0 | | | | | | | | | | |
| Auto-select (9) | Manufacturer ID | 4 | 555 | AA | 2AA | 55 | [BA]555 | 90 | [BA]X00 | 0001 | | | | |
| | Device ID (10) | 6 | 555 | AA | 2AA | 55 | [BA]555 | 90 | [BA]X01 | 227E | [BA]X0E | (10) | [BA]X0F | 2200 |
| | Indicator Bits | 4 | 555 | AA | 2AA | 55 | [BA]555 | 90 | [BA]X03 | (11) | | | | |
| Program | | 4 | 555 | AA | 2AA | 55 | 555 | A0 | PA | Data | | | | |
| Write to Buffer (17) | | 6 | 555 | AA | 2AA | 55 | SA | 25 | SA | WC | PA | PD | WBL | PD |
| Program Buffer to Flash | | 1 | SA | 29 | | | | | | | | | | |
| Write to Buffer Abort Reset (17) | | 3 | 555 | AA | 2AA | 55 | 555 | F0 | | | | | | |
| Chip Erase | | 6 | 555 | AA | 2AA | 55 | 555 | 80 | 555 | AA | 2AA | 55 | 555 | 10 |
| Sector Erase | | 6 | 555 | AA | 2AA | 55 | 555 | 80 | 555 | AA | 2AA | 55 | SA | 30 |
| Program/Erase Suspend (14) | | 1 | BA | B0 | | | | | | | | | | |
| Program/Erase Resume (15) | | 1 | BA | 30 | | | | | | | | | | |
| CFI Query (16) | | 1 | [BA]555 | 98 | | | | | | | | | | |
| Unlock Bypass Mode | Unlock Bypass Entry | 3 | 555 | AA | 2AA | 55 | 555 | 20 | | | | | | |
| | Unlock Bypass Program (12, 13) | 2 | XX | A0 | PA | PD | | | | | | | | |
| | Unlock Bypass Sector Erase (12, 13) | 2 | XX | 80 | SA | 30 | | | | | | | | |
| | Unlock Bypass Erase (12, 13) | 2 | XX | 80 | XXX | 10 | | | | | | | | |
| | Unlock Bypass CFI (12, 13) | 1 | BA | 98 | | | | | | | | | | |
| | Unlock Bypass Reset | 2 | XX | 90 | XXX | 00 | | | | | | | | |
| Secured Silicon Sector Command Definitions | | | | | | | | | | | | | | |
| Secured Silicon Sector | Secured Silicon Sector Entry (18) | 3 | 555 | AA | 2AA | 55 | 555 | 88 | | | | | | |
| | Secured Silicon Sector Program | 2 | XX | A0 | PA | data | | | | | | | | |
| | Secured Silicon Sector Read | 1 | RA | data | | | | | | | | | | |
| | Secured Silicon Sector Exit (19) | 4 | 555 | AA | 2AA | 55 | 555 | 90 | XX | 00 | | | | |

***Legend***

*X = Don't care.*

*RA = Read Address.*

*RD = Read Data.*

*PA = Address of the memory location to be programmed. Addresses latch on the falling edge of the WE# or CE# pulse whichever happens later.*

*PD = Program Data. Data latches on the rising edge of WE# or CE# pulse, whichever occurs first.*

*SA = Sector Address. PL127/129N = A22 – A15;*

*PL256N = A23 – A15.*

*BA = Bank Address. PL256N = A23 – A21; PL127N = A22 – A20; PL127N = A21 – A20.*

*WBL = Write Buffer Location. Address must be within the same write buffer page as PA.*

*WC = Word Count. Number of write buffer locations to load minus 1.*

***Notes***

1. *See Table 7.1 on page 20 for description of bus operations.*
2. *All values are in hexadecimal.*
3. *Except for the following, all bus cycles are write cycle: read cycle, fourth through sixth cycles of the Autoselect commands, fourth cycle of the password verify command, and any cycle reading at RD(0) and RD(1).*
4. *Data bits DQ15 – DQ8 are don't care in command sequences, except for RD, PD, WD, PWD, and PWD3 – PWD0.*
5. *Unless otherwise noted, these address bits are* don't cares: *PL127: A22 – A15; 129N: A21 – A15; PL256N: A23 – A14.*
6. *Writing incorrect address and data values or writing them in the improper sequence may place the device in an unknown state. The system must write the reset command to return the device to reading array data.*
7. *No unlock or command cycles required when bank is reading array data.*

8. *The Reset command is required to return to reading array data (or to the erase-suspend-read mode if previously in Erase Suspend) when a bank is in the autoselect mode, or if DQ5 goes high (while the bank is providing status information) or performing sector lock/unlock.*
9. *The fourth cycle of the autoselect command sequence is a read cycle. The system must provide the bank address.* *See Autoselect on page 23.*
10. *Device IDs: PL256N = 223Ch; PL127N = 2220h; PL129N = 2221h.*
11. *See Autoselect on page 23.*
12. *The Unlock Bypass command sequence is required prior to this command sequence.*
13. *The Unlock Bypass Reset command is required to return to reading array data when the bank is in the unlock bypass mode.*
14. *The system may read and program in non-erasing sectors, or enter the autoselect mode, when in the Erase Suspend mode. The Erase Suspend command is valid only during a sector erase operation, and requires the bank address.*
15. *The Erase Resume command is valid only during the Erase Suspend mode, and requires the bank address.*
16. *The total number of cycles in the command sequence is determined by the number of words written to the write buffer. The maximum number of cycles in the command sequence is 37.*
17. *Command sequence resets device for next command after write-to-buffer operation.*
18. *Entry commands are needed to enter a specific mode to enable instructions only available within that mode.*
19. *The Exit command must be issued to reset the device into read mode. Otherwise the device hangs.*
20. *The following mode cannot be performed at the same time. Autoselect/CFI/Unlock Bypass/Secured Silicon. Command sequence resets device for next command after write-to-buffer operation.*
21. *Command is valid when device is ready to read array data or when device is in autoselect mode. Address equals 55h on all future devices, but 555h for PL256N.*
22. *Requires Entry command sequence prior to execution. Secured Silicon Sector Exit Reset command is required to exit this mode; device may otherwise be placed in an unknown state.*

**Table 12.2** Sector Protection Commands

| Command Sequence (Notes) | | Cycles | Bus Cycles (Notes 1 – 6) | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | First | | Second | | Third | | Fourth | | Fifth | | Sixth | | Seventh | |
| | | | Addr | Data | Addr | Data | Addr | Data | Addr | Data | Addr | Data | Addr | Data | Addr | Data |
| Lock Register Command Set Definitions | | | | | | | | | | | | | | | | |
| Lock Register | Lock Register Command Set Entry (25) | 3 | 555 | AA | 2AA | 55 | 555 | 40 | | | | | | | | |
| | Lock Register Bits Program (26) | 2 | XX | A0 | 00 | data | | | | | | | | | | |
| | Lock Register Bits Read | 1 | 00 | data | | | | | | | | | | | | |
| | Lock Register Command Set Exit (27) | 2 | XX | 90 | XX | 00 | | | | | | | | | | |
| Password Protection Command Set Definitions | | | | | | | | | | | | | | | | |
| Password | Password Protection Command Set Entry (25) | 3 | 555 | AA | 2AA | 55 | 555 | 60 | | | | | | | | |
| | Password Program | 2 | XX | A0 | 00/01 02/03 | PWD0/ PWD1/ PWD2/ PWD3 | | | | | | | | | | |
| | Password Read | 4 | 00 | PWD 0 | 01 | PWD1 | 02 | PWD2 | 03 | PWD3 | | | | | | |
| | Password Unlock | 7 | 00 | 25 | 00 | 03 | 00 | PWD0 | 01 | PWD1 | 02 | PWD2 | 03 | PWD3 | 00 | 29 |
| | Password Protection Command Set Exit (27) | 2 | XX | 90 | XX | 00 | | | | | | | | | | |
| Non-Volatile Sector Protection Command Set Definitions | | | | | | | | | | | | | | | | |
| PPB | Non-Volatile Sector Protection Command Set Entry (25) | 3 | 555 | AA | 2AA | 55 | [BA]555 | C0 | | | | | | | | |
| | PPB Program | 2 | XX | A0 | [BA]SA | 00 | | | | | | | | | | |
| | All PPB Erase (22) | 2 | XX | 80 | 00 | 30 | | | | | | | | | | |
| | PPB Status Read | 1 | [BA]SA | RD(0) | | | | | | | | | | | | |
| | Non-Volatile Sector Protection Command Set Exit (27) | 2 | XX | 90 | XX | 00 | | | | | | | | | | |
| Global Non-Volatile Sector Protection Freeze Command Set Definitions | | | | | | | | | | | | | | | | |
| PPB Lock Bit | Global Volatile Sector Protection Freeze Command Set Entry (25) | 3 | 555 | AA | 2AA | 55 | 555 | 50 | | | | | | | | |
| | PPB Lock Bit Set | 2 | XX | A0 | XX | 00 | | | | | | | | | | |
| | PPB Lock Bit Status Read | 1 | BA | RD(0) | | | | | | | | | | | | |
| | Global Volatile Sector Protection Freeze Command Set Exit (27) | 2 | XX | 90 | XX | 00 | | | | | | | | | | |
| Volatile Sector Protection Command Set Definitions | | | | | | | | | | | | | | | | |
| DYB | Volatile Sector Protection Command Set Entry (25) | 3 | 555 | AA | 2AA | 55 | [BA]555 | E0 | | | | | | | | |
| | DYB Set | 2 | XX | A0 | [BA]SA | 00 | | | | | | | | | | |
| | DYB Clear | 2 | XX | A0 | [BA]SA | 01 | | | | | | | | | | |
| | DYB Status Read | 1 | [BA]SA | RD(0) | | | | | | | | | | | | |
| | Volatile Sector Protection Command Set Exit (27) | 2 | XX | 90 | XX | 00 | | | | | | | | | | |

***Legend***

*X = Don't care*

*RA = Read Address.*

*RD = Read Data.*

*PA = Address of the memory location to be programmed. Addresses latch on the falling edge of the WE# or CE# pulse whichever happens later.*

*PD = Program Data. Data latches on the rising edge of WE# or CE# pulse, whichever occurs first.*

*SA = Sector Address. PL127/129N = A22 – A15; PL256N = A23 – A15*

*BA = Bank Address. PL256N = A23 – A21; PL127N = A22 – A20; PL127N = A21 – A20.*

*WBL = Write Buffer Location. Address must be within the same write buffer page as PA.*

*WC = Word Count. Number of write buffer locations to load minus 1.*

*PWD3 – PWD0 = Password Data. PD3 – PD0 present four 16 bit combinations that represent the 64-bit Password*

*RD(0) = DQ0 protection indicator bit. If protected, DQ0 = 0, if unprotected, DQ0 = 1.*

***Notes***

1. *See Table 7.1 on page 20 for description of bus operations.*
2. *All values are in hexadecimal.*
3. *Except for the following, all bus cycles are write cycle: read cycle, fourth through sixth cycles of the Autoselect commands, and password verify commands, and any cycle reading at RD(0) and RD(1).*
4. *Data bits DQ15 – DQ8 are don't care in command sequences, except for RD, PD, WD, PWD, and PWD3 – PWD0.*
5. *Unless otherwise noted, these address bits are* don't cares: *PL127: A22 – A15; 129N: A21 – A15; PL256N: A23 – A14.*
6. *Writing incorrect address and data values or writing them in the improper sequence may place the device in an unknown state. The system must write the reset command to return the device to reading array data.*
7. *No unlock or command cycles required when bank is reading array data.*
8. *The Reset command is required to return to reading array data (or to the erase-suspend-read mode if previously in Erase Suspend) when a bank is in the autoselect mode, or if DQ5 goes high (while the bank is providing status information) or performing sector lock/unlock.*
9. *The fourth cycle of the autoselect command sequence is a read cycle. The system must provide the bank address. See Autoselect on page 23.*
10. *The data is 0000h for an unlocked sector and 0001h for a locked sector.*
11. *Device IDs: PL256N = 223Ch; PL127N = 2220h;*
    *PL129N = 2221h.*
12. *See Autoselect on page 23.*
13. *The Unlock Bypass command sequence is required prior to this command sequence.*
14. *The Unlock Bypass Reset command is required to return to reading array data when the bank is in the unlock bypass mode.The system may read and program in non-erasing sectors, or enter the autoselect mode, when in the Erase Suspend mode. The Erase Suspend command is valid only during a sector erase operation, and requires the bank address.*
15. *The Erase Resume command is valid only during the Erase Suspend mode, and requires the bank address.*
16. *Command is valid when device is ready to read array data or when device is in autoselect mode.The total number of cycles in the command sequence is determined by the number of words written to the write buffer. The maximum number of cycles in the command sequence is 37.*
17. *The entire four bus-cycle sequence must be entered for which portion of the password.*
18. *The Unlock Bypass Reset command is required to return to reading array data when the bank is in the unlock bypass mode.The system may read and program in non-erasing sectors, or enter the autoselect mode, when in the Erase Suspend mode. The Erase Suspend command is valid only during a sector erase operation, and requires the bank address.*
19. *The Erase Resume command is valid only during the Erase Suspend mode, and requires the bank address.*
20. *Command is valid when device is ready to read array data or when device is in autoselect mode.The total number of cycles in the command sequence is determined by the number of words written to the write buffer. The maximum number of cycles in the command sequence is 37.*
21. *The entire four bus-cycle sequence must be entered for which portion of the password.*
22. *The ALL PPB ERASE command pre-programs all PPBs before erasure to prevent over-erasure of PPBs.*
23. *WP#/ACC must be at VHH during the entire operation of this command.*
24. *Command sequence resets device for next command after write-to-buffer operation.*
25. *Entry commands are needed to enter a specific mode to enable instructions only available within that mode.*
26. *If both the Persistent Protection Mode Locking Bit and the password Protection Mode Locking Bit are set a the same time, the command operation aborts and returns the device to the default Persistent Sector Protection Mode.*
27. *The Exit command must be issued to reset the device into read mode. Otherwise the device hangs.*

# 13. Common Flash Memory Interface

The Common Flash Interface (CFI) specification outlines device and host system software interrogation handshake, which allows specific vendor-specified soft-ware algorithms to be used for entire families of devices. Software support can then be device-independent, JEDEC ID-independent, and forward- and backward-compatible for the specified flash device families. Flash vendors can standardize their existing interfaces for long-term compatibility.

This device enters the CFI Query mode when the system writes the CFI Query command, 98h, to address (BA)555h any time the device is ready to read array data. The system can read CFI information at the addresses given in Tables 12.3 – 12.6) within that bank. All reads outside of the CFI address range, within the bank, return non-valid data. Reads from other banks are allowed, writes are not. To terminate reading CFI data, the system must write the reset command.

The following is a C source code example of using the CFI Entry and Exit functions. Refer to the *Spansion Low Level Driver User's Guide* (available at www.spansion.com) for general information on Spansion Flash memory software development guidelines.

```
/* Example: CFI Entry command */
 *((UINT16 *)bank_addr + 0x555) = 0x0098;   /* write CFI entry command         */

/* Example: CFI Exit command */
 *((UINT16 *)bank_addr + 0x000) = 0x00F0;   /* write cfi exit command          */
```

For further information, please see the CFI Specification (see JEDEC publications JEP137-A and JESD68.01and CFI Publication 100). Please contact your sales office for copies of these documents.

**Table 13.1** CFI Query Identification String

| Addresses | Data | Description |
|---|---|---|
| 10h<br>11h<br>12h | 0051h<br>0052h<br>0059h | Query Unique ASCII string *QRY* |
| 13h<br>14h | 0002h<br>0000h | Primary OEM Command Set |
| 15h<br>16h | 0040h<br>0000h | Address for Primary Extended Table |
| 17h<br>18h | 0000h<br>0000h | Alternate OEM Command Set (00h = none exists) |
| 19h<br>1Ah | 0000h<br>0000h | Address for Alternate OEM Extended Table (00h = none exists) |

**Table 13.2** System Interface String

| Addresses | Data | Description |
|---|---|---|
| 1Bh | 0027h | $V_{CC}$ Min. (write/erase)<br>D7 – D4: volt, D3 – D0: 100 millivolt |
| 1Ch | 0036h | $V_{CC}$ Max. (write/erase)<br>D7 – D4: volt, D3 – D0: 100 millivolt |
| 1Dh | 0000h | $V_{PP}$ Min. voltage (00h = no $V_{PP}$ pin present) |
| 1Eh | 0000h | $V_{PP}$ Max. voltage (00h = no $V_{PP}$ pin present) |
| 1Fh | 0006h | Typical timeout per single byte/word write $2^N$ µs |
| 20h | 0009h | Typical timeout for Min. size buffer write $2^N$ µs (00h = not supported) |
| 21h | 000Bh | Typical timeout per individual block erase $2^N$ ms |
| 22h | 0000h | Typical timeout for full chip erase $2^N$ ms (00h = not supported) |
| 23h | 0003h | Max. timeout for byte/word write $2^N$ times typical |
| 24h | 0003h | Max. timeout for buffer write $2^N$ times typical |
| 25h | 0002h | Max. timeout per individual block erase $2^N$ times typical |
| 26h | 0000h | Max. timeout for full chip erase $2^N$ times typical (00h = not supported) |

**Table 13.3** Device Geometry Definition

| Addresses | Data | Description |
|---|---|---|
| 27h | 0019h (PL256N)<br>0018h (PL127N)<br>0018h (PL129N) | Device Size = $2^N$ byte |
| 28h<br>29h | 0001h<br>0000h | Flash Device Interface description (see CFI publication 100) |
| 2Ah<br>2Bh | 0006h<br>0000h | Max. number of byte in multi-byte write = $2^N$<br>(00h = not supported) |
| 2Ch | 0003h | Number of Erase Block Regions within device |
| 2Dh<br>2Eh<br>2Fh<br>30h | 0003h<br>0000h<br>0000h<br>0001h | Erase Block Region 1 Information<br>(see the CFI specification or CFI publication 100) |
| 31h | 007Dh (PL256N)<br>003Dh (PL127N)<br>003Dh (PL129N) | Erase Block Region 2 Information<br>(see the CFI specification or CFI publication 100) |
| 32h<br>33h<br>34h | 0000h<br>0000h<br>0004h | |
| 35h<br>36h<br>37h<br>38h | 0003h<br>0000h<br>0000h<br>0001h | Erase Block Region 3 Information<br>(see the CFI specification or CFI publication 100) |

**Table 13.4** Primary Vendor-Specific Extended Query

| Addresses | Data | Description |
|---|---|---|
| 40h<br>41h<br>42h | 0050h<br>0052h<br>0049h | Query-unique ASCII string *PRI* |
| 43h | 0031h | Major version number, ASCII (reflects modifications to the silicon) |
| 44h | 0034h | Minor version number, ASCII (reflects modifications to the CFI table) |
| 45h | 0010h | Address Sensitive Unlock (Bits 1 – 0)<br>0 = Required, 1 = Not Required<br>Silicon Technology (Bits 5 – 2) 0100 = 0.11 µm |
| 46h | 0002h | Erase Suspend<br>0 = Not Supported, 1 = To Read Only, 2 = To Read & Write |
| 47h | 0001h | Sector Protect<br>0 = Not Supported, X = Number of sectors in per group |
| 48h | 0000h | Sector Temporary Unprotect<br>00 = Not Supported, 01 = Supported |
| 49h | 0008h (PL-N) | Sector Protect/Unprotect scheme<br>01 =29F040 mode, 02 = 29F016 mode, 03 = 29F400 mode,<br>04 = 29LV800 mode 07 = New Sector Protect mode,<br>08 = Advanced Sector Protection |
| 4Ah | 0073h (PL256N)<br>003Bh (PL127N)<br>003Bh (PL129N) | Simultaneous Operation<br>00 = Not Supported, X = Number of Sectors except Bank A |
| 4Bh | 0000h | Burst Mode Type<br>00 = Not Supported, 01 = Supported |
| 4Ch | 0002h (PL-N) | Page Mode Type<br>00 = Not Supported, 01 = 4 Word Page, 02 = 8 Word Page |
| 4Dh | 0085h | ACC (Acceleration) Supply Minimum<br>00h = Not Supported, D7 – D4: Volt, D3 – D0: 100 mV |
| 4Eh | 0095h | ACC (Acceleration) Supply Maximum<br>00h = Not Supported, D7 – D4: Volt, D3 – D0: 100 mV |
| 4Fh | 0001h | Top/Bottom Boot Sector Flag<br>00h = No Boot, 01h = Dual Boot Device, 02h = Bottom Boot Device, 03h = Top Boot Device |
| 50h | 0001h | Program Suspend<br>0 = Not supported, 1 = Supported |
| 51h | 0001h | Unlock Bypass<br>00 = Not Supported, 01=Supported |
| 52h | 0007h | Secured Silicon Sector (Customer OTP Area) Size $2^N$ bytes |
| 53h | 000Fh | Hardware Reset Low Time-out during an embedded algorithm to read mode Maximum $2^N$ ns |
| 54h | 000Eh | Hardware Reset Low Time-out not during an embedded algorithm to read mode Maximum $2^N$ ns |
| 55h | 0005h | Erase Suspend Time-out Maximum $2^N$ µs |
| 56h | 0005h | Program Suspend Time-out Maximum $2^N$ µs |
| 57h | 0004h | Bank Organization<br>00 = Data at 4Ah is zero, X = Number of Banks |
| 58h | 0013h (PL256N)<br>000Bh (PL127N)<br>000Bh (PL129N) | Bank A Region Information. X = Number of sectors in bank |
| 59h | 0030h (PL256N)<br>0018h (PL127N)<br>0018h (PL129N) | Bank 1 Region Information. X = Number of sectors in bank |
| 5Ah | 0030h (PL256N)<br>0018h (PL127N)<br>0018h (PL129N) | Bank 2 Region Information. X = Number of sectors in bank |
| 5Bh | 0013h (PL256N)<br>000Bh (PL127N)<br>000Bh (PL129N) | Bank 3 Region Information. X = Number of sectors in bank |

# 14. Revision History

| Section | Description |
|---|---|
| **Revision A0 (February 28, 2005)** | |
| | Initial release |
| **Revision A1 (August 8, 2005)** | |
| Performance Characteristics | Updated Package Options |
| MCP Look-Ahead Connection Diagram | Corrected pinout |
| Memory Map | Added Sector and Memory Address Map for S29PL127N |
| Device Operation Table | Added Dual Chip Enable Device Operation Table |
| $V_{CC}$ Power Up | Updated $t_{VCS}$.<br>Added $V_{CC}$ ramp rate restriction |
| DC Characteristics | Updated typical and maximum values. |
| **Revision A2 (October 25, 2005)** | |
| Ordering Information | Updated table |
| Connection Diagram and Package Dimensions - S29PL-N Fortified BGA | Added pinout and package dimensions. |
| Global | Changed data sheet status from Advance Information to Preliminary.<br>Removed Byte Address Information |
| Distinctive and Performance Characteristics | Removed Enhanced VersatileI/O, updated read access times, and Package options. |
| Logic Symbol and Block Diagram | Removed $V_{IO}$ from Logic Symbol and Block Diagram. |
| Erase and Programming Performance | Updated table |
| Write Buffer Programming | Updated Write Buffer Abort Description. |
| Operating Ranges | Updated $V_{IO}$ supply voltages. |
| DC characteristics | Updated $I_{CC1}$, $I_{CC4}$, $I_{CC6}$. |
| **Revision A3 (November 14, 2005)** | |
| Ordering Information | Updated table |
| Valid Combinations Table | Updated table |
| **Revision A4 (November 23, 2005)** | |
| Logic Symbols | Removed $V_{IO}$ from the illustrations |
| Block Diagram | Removed $V_{IO}$ from the illustration |
| Connection Diagrams | Modified Fortified BGA Pinout (LAA064) |
| PL129N Sector and Memory Address Map | Updated Address Ranges for Banks 2A and 2B |
| **Revision A5 (June 6, 2007)** | |
| Global | Removed LAA064 package offering and all relevant ordering information |

### *Colophon*

The products described in this document are designed, developed and manufactured as contemplated for general use, including without limitation, ordinary industrial use, general office use, personal use, and household use, but are not designed, developed and manufactured as contemplated (1) for any use that includes fatal risks or dangers that, unless extremely high safety is secured, could have a serious effect to the public, and could lead directly to death, personal injury, severe physical damage or other loss (i.e., nuclear reaction control in nuclear facility, aircraft flight control, air traffic control, mass transport control, medical life support system, missile launch control in weapon system), or (2) for any use where chance of failure is intolerable (i.e., submersible repeater and artificial satellite). Please note that Spansion will not be liable to you and/or any third party for any claims or damages arising in connection with above-mentioned uses of the products. Any semiconductor devices have an inherent chance of failure. You must protect against injury, damage or loss from such failures by incorporating safety design measures into your facility and equipment such as redundancy, fire protection, and prevention of over-current levels and other abnormal operating conditions. If any products described in this document represent goods or technologies subject to certain restrictions on export under the Foreign Exchange and Foreign Trade Law of Japan, the US Export Administration Regulations or the applicable laws of any other country, the prior authorization by the respective government entity will be required for export of those products.

### *Trademarks and Notice*

The contents of this document are subject to change without notice. This document may contain information on a Spansion product under development by Spansion. Spansion reserves the right to change or discontinue work on any product without notice. The information in this document is provided as is without warranty or guarantee of any kind as to its accuracy, completeness, operability, fitness for particular purpose, merchantability, non-infringement of third-party rights, or any other warranty, express, implied, or statutory. Spansion assumes no liability for any damages of any kind arising out of the use of the information in this document.