Chipsmall

Chipsmall Limited consists of a professional team with an average of over 10 year of expertise in the distribution of electronic components. Based in Hongkong, we have already established firm and mutual-benefit business relationships with customers from,Europe,America and south Asia,supplying obsolete and hard-to-find components to meet their specific needs.

With the principle of "Quality Parts,Customers Priority,Honest Operation,and Considerate Service",our business mainly focus on the distribution of electronic components. Line cards we deal with include Microchip,ALPS,ROHM,Xilinx,Pulse,ON,Everlight and Freescale. Main products comprise IC,Modules,Potentiometer,IC Socket,Relay,Connector.Our parts cover such applications as commercial,industrial, and automotives areas.

We are looking forward to setting up business relationship with you and hope to provide you with the best service and solution. Let us make a better world for our industry!

Contact us

Tel: +86-755-8981 8866 Fax: +86-755-8427 6832
Email & Skype: info@chipsmall.com Web: www.chipsmall.com
Address: A1208, Overseas Decoration Building, #122 Zhenhua RD., Futian, Shenzhen, China

AT45DQ161

**16-Mbit DataFlash, 2.3V Minimum
SPI Serial Flash Memory with Dual-I/O and Quad-I/O Support**

Features

- Single 2.3V - 3.6V supply
- Serial Peripheral Interface (SPI) compatible
  - Supports SPI modes 0 and 3
  - Supports RapidS™ operation
  - Supports Dual-input and Quad-input Buffer Write
  - Supports Dual-output and Quad-output Read
- Very high operating frequencies
  - 85MHz (for SPI)
  - 85MHz (for Dual-I/O and Quad-I/O)
  - Clock-to-output time (t_V) of 6ns maximum
- User configurable page size
  - 512 bytes per page
  - 528 bytes per page (default)
  - Page size can be factory pre-configured for 512 bytes
- Two fully independent SRAM data buffers (512/528 bytes)
  - Allows receiving data while reprogramming the main memory array
- Flexible programming options
  - Byte/Page Program (1 to 512/528 bytes) directly into main memory
  - Buffer Write
  - Buffer to Main Memory Page Program
- Flexible erase options
  - Page Erase (512/528 bytes), Block Erase (4KB)
  - Sector Erase (128KB), Chip Erase (16-Mbits)
- Program and Erase Suspend/Resume
- Advanced hardware and software data protection features
  - Individual sector protection
  - Individual sector lockdown to make any sector permanently read-only
- 128-byte, One-Time Programmable (OTP) Security Register
  - 64 bytes factory programmed with a unique identifier
  - 64 bytes user programmable
- Hardware and software controlled reset options
- JEDEC Standard Manufacturer and Device ID Read
- Low-power dissipation
  - 500nA Ultra-Deep Power-Down current (typical)
  - 3µA Deep Power-Down current (typical)
  - 25µA Standby current (typical)
  - 11mA Active Read current (typical at 20MHz)
- Endurance: 100,000 program/erase cycles per page minimum (50,000 cycles for extended temperature option)
- Data retention: 20 years
- Complies with full industrial temperature range (extended temperature optional)
- Green (Pb/Halide-free/RoHS compliant) packaging options
  - 8-lead SOIC (0.150" wide and 0.208" wide)
  - 8-pad Ultra-thin DFN (5 x 6 x 0.6mm)
  - 9-ball Ultra-thin UBGA (6 x 6 x 0.6mm)

Description

The AT45DQ161 is a 2.3V minimum, serial-interface sequential access Flash memory ideally suited for a wide variety of digital voice, image, program code, and data storage applications. The AT45DQ161 also supports Dual-I/O, Quad-I/O and the RapidS serial interface for applications requiring very high speed operation. Its 17,301,504 bits of memory are organized as 4,096 pages of 512 bytes or 528 bytes each. In addition to the main memory, the AT45DQ161 also contains two SRAM buffers of 512/528 bytes each. The buffers allow receiving of data while a page in the main memory is being reprogrammed. Interleaving between both buffers can dramatically increase a system's ability to write a continuous data stream. In addition, the SRAM buffers can be used as additional system scratch pad memory, and E^2PROM emulation (bit or byte alterability) can be easily handled with a self-contained three step read-modify-write operation.

Unlike conventional Flash memories that are accessed randomly with multiple address lines and a parallel interface, the DataFlash® uses a serial interface to sequentially access its data. The simple sequential access dramatically reduces active pin count, facilitates simplified hardware layout, increases system reliability, minimizes switching noise, and reduces package size. The device is optimized for use in many commercial and industrial applications where high-density, low-pin count, low-voltage, and low-power are essential.

To allow for simple in-system re-programmability, the AT45DQ161 does not require high input voltages for programming. The device operates from a single 2.3V to 3.6V power supply for the erase and program and read operations. The AT45DQ161 is enabled through the Chip Select pin ($\overline{CS}$) and accessed via a 3-wire interface consisting of the Serial Input (SI), Serial Output (SO), and the Serial Clock (SCK).

All programming and erase cycles are self-timed.

1. Pin Configurations and Pinouts

Figure 1-1. Pinouts
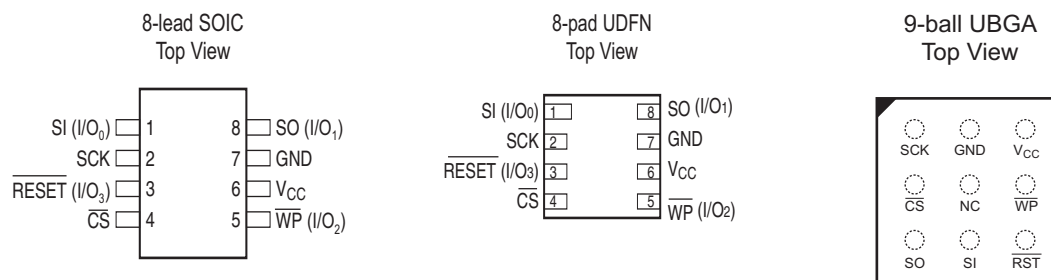
Note: 1. The metal pad on the bottom of the UDFN package is not internally connected to a voltage potential. This pad can be a "no connect" or connected to GND.
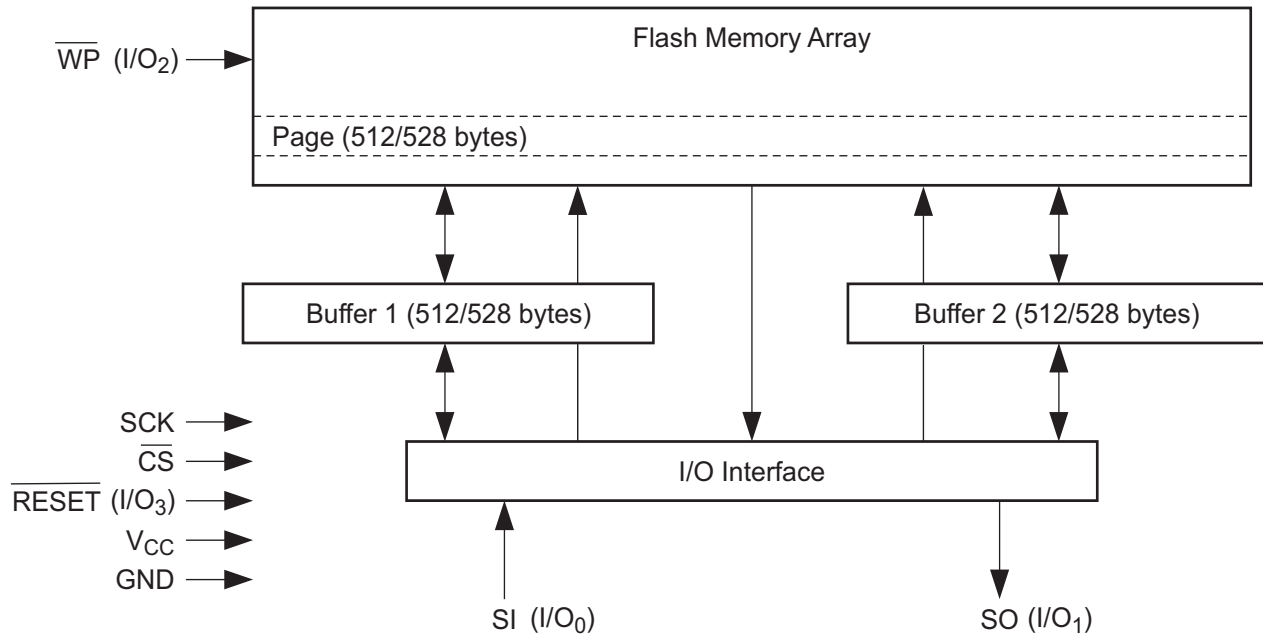
Table 1-1.   Pin Configurations

Symbol	Name and Function	Asserted State	Type
$\overline{CS}$	**Chip Select:** Asserting the $\overline{CS}$ pin selects the device. When the $\overline{CS}$ pin is deasserted, the device will be deselected and normally be placed in the standby mode (not Deep Power-Down mode) and the output pin (SO) will be in a high-impedance state. When the device is deselected, data will not be accepted on the input pin (SI). A high-to-low transition on the $\overline{CS}$ pin is required to start an operation and a low-to-high transition is required to end an operation. When ending an internally self-timed operation such as a program or erase cycle, the device will not enter the standby mode until the completion of the operation.	Low	Input
SCK	**Serial Clock:** This pin is used to provide a clock to the device and is used to control the flow of data to and from the device. Command, address, and input data present on the SI pin is always latched on the rising edge of SCK, while output data on the SO pin is always clocked out on the falling edge of SCK.	—	Input
SI (I/O$_0$)	**Serial Input (I/O$_0$):** The SI pin is used to shift data into the device. The SI pin is used for all data input including command and address sequences. Data on the SI pin is always latched on the rising edge of SCK. With the Dual-output and Quad-output Read Array commands, the SI pin becomes an output pin (I/O$_0$) and, along with other pins, allows two bits (on I/O$_{1-0}$) or four bits (on I/O$_{3-0}$) of data to be clocked out on every falling edge of SCK. To maintain consistency with SPI nomenclature, the SI (I/O$_0$) pin will be referenced as SI throughout the document with exception to sections dealing with the Dual-output and Quad-output Read Array commands in which it will be referenced as I/O$_0$. Data present on the SI pin will be ignored whenever the device is deselected ($\overline{CS}$ is deasserted).	—	Input/ Output
SO (I/O$_1$)	**Serial Output (I/O$_1$):** The SO pin is used to shift data out from the device. Data on the SO pin is always clocked out on the falling edge of SCK. With the Dual-output and Quad-output Read Array commands, the SO pin is used as an output pin (I/O$_1$) in conjunction with other pins to allow two bits (on I/O$_{1-0}$) or four bits (on I/O$_{3-0}$) of data to be clocked out on every falling edge of SCK. To maintain consistency with SPI nomenclature, the SO (I/O$_1$) pin will be referenced as SO throughout the document with exception to sections dealing with the Dual-output and Quad-output Read Array commands in which it will be referenced as I/O$_1$. The SO pin will be in a high-impedance state whenever the device is deselected ($\overline{CS}$ is deasserted).	—	Input/ Output

adesto®
TECHNOLOGIES

Table 1-1. Pin Configurations (Continued)

Symbol	Name and Function	Asserted State	Type
$\overline{WP}$ (I/O$_2$)	**Write Protect (I/O$_2$):** When the $\overline{WP}$ pin is asserted, all sectors specified for protection by the Sector Protection Register will be protected against program and erase operations regardless of whether the Enable Sector Protection command has been issued or not. The $\overline{WP}$ pin functions independently of the software controlled protection method. After the $\overline{WP}$ pin goes low, the contents of the Sector Protection Register cannot be modified. The $\overline{WP}$ pin must be driven at all times or pulled-high using an external pull-up resistor. If a program or erase command is issued to the device while the $\overline{WP}$ pin is asserted, the device will simply ignore the command and perform no operation. The device will return to the idle state once the $\overline{CS}$ pin has been deasserted. The Enable Sector Protection command and the Sector Lockdown command, however, will be recognized by the device when the $\overline{WP}$ pin is asserted. The $\overline{WP}$ pin is internally pulled-high and may be left floating if hardware controlled protection will not be used. However, it is recommended that the $\overline{WP}$ pin also be externally connected to V_{CC} whenever possible. With the Quad-output Read Array command, the $\overline{WP}$ pin becomes an output pin (I/O$_2$) and, when used with other pins, allows four bits (on I/O$_{3-0}$) of data to be clocked out on every falling edge of SCK. The QE bit in the Configuration Register must be set in order for the $\overline{WP}$ pin to be used as an I/O data pin.	Low	Input/ Output
$\overline{RESET}$ (I/O$_3$)	**Reset (I/O$_3$):** A low state on the reset pin ($\overline{RESET}$) will terminate the operation in progress and reset the internal state machine to an idle state. The device will remain in the reset condition as long as a low level is present on the $\overline{RESET}$ pin. Normal operation can resume once the $\overline{RESET}$ pin is brought back to a high level. With the Quad-output Read Array command, the $\overline{RESET}$ pin becomes an output pin (I/O$_3$) and, when used with other pins, allows four bits (on I/O$_{3-0}$) of data to be clocked out on every falling edge of SCK. The QE bit in the Configuration Register must be set in order for the $\overline{RESET}$ pin to be used as an I/O data pin. The device incorporates an internal power-on reset circuit, so there are no restrictions on the $\overline{RESET}$ pin during power-on sequences. If this pin and feature is not utilized, then it is recommended that the $\overline{RESET}$ pin be driven high externally.	Low	Input/ Output
V_{CC}	**Device Power Supply:** The V_{CC} pin is used to supply the source voltage to the device. Operations at invalid V_{CC} voltages may produce spurious results and should not be attempted.	—	Power
GND	**Ground:** The ground reference for the power supply. GND should be connected to the system ground.	—	Ground

adesto
TECHNOLOGIES

2.   Block Diagram

Figure 2-1.   Block Diagram

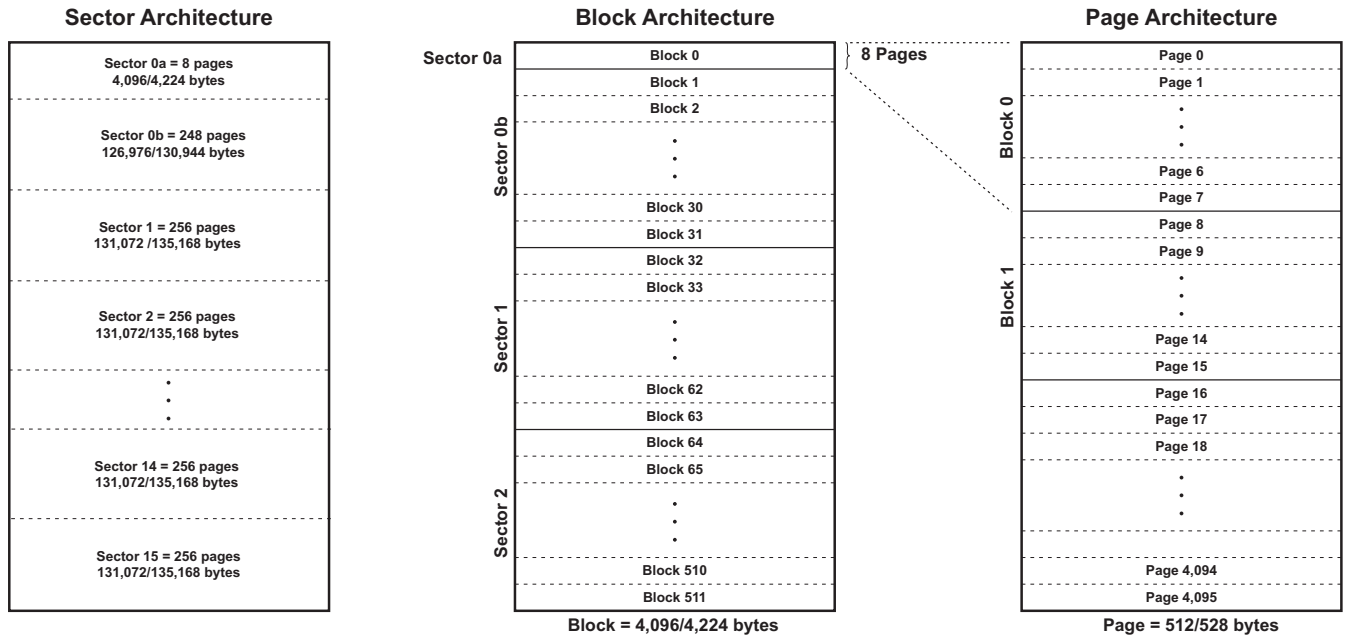Note: I/O_{3-0} pin naming convention is used for Dual-I/O and Quad-I/O commands.

3. Memory Array

To provide optimal flexibility, the AT45DQ161 memory array is divided into three levels of granularity comprising of sectors, blocks, and pages. <blue>Figure 3-1, Memory Architecture Diagram illustrates the breakdown of each level and details the number of pages per sector and block. Program operations to the DataFlash can be done at the full page level or at the byte level (a variable number of bytes). The erase operations can be performed at the chip, sector, block, or page level.

Figure 3-1.   Memory Architecture Diagram

Sector Architecture

Sector 0a = 8 pages 4,096/4,224 bytes
Sector 0b = 248 pages 126,976/130,944 bytes
Sector 1 = 256 pages 131,072 /135,168 bytes
Sector 2 = 256 pages 131,072/135,168 bytes
• • •
Sector 14 = 256 pages 131,072/135,168 bytes
Sector 15 = 256 pages 131,072/135,168 bytes

Block Architecture

Sector 0a — Block 0 } 8 Pages

Sector 0b:
- Block 1
- Block 2
- •••
- Block 30
- Block 31

Sector 1:
- Block 32
- Block 33
- •••
- Block 62
- Block 63

Sector 2:
- Block 64
- Block 65
- •••
- Block 510
- Block 511

Block = 4,096/4,224 bytes

Page Architecture

Block 0:
- Page 0
- Page 1
- •••
- Page 6
- Page 7

Block 1:
- Page 8
- Page 9
- •••
- Page 14
- Page 15
- Page 16
- Page 17
- Page 18
- •••
- Page 4,094
- Page 4,095

Page = 512/528 bytes

adesto
TECHNOLOGIES

4.   Device Operation

The device operation is controlled by instructions from the host processor. The list of instructions and their associated opcodes are contained in <blue>Table 15-1 on page 47 through <blue>Table 15-4 on page 48. A valid instruction starts with the falling edge of $\overline{CS}$ followed by the appropriate 8-bit opcode and the desired buffer or main memory address location. While the $\overline{CS}$ pin is low, toggling the SCK pin controls the loading of the opcode and the desired buffer or main memory address location through the SI (Serial Input) pin. All instructions, addresses, and data are transferred with the Most Significant
Bit (MSB) first.

Three address bytes are used to address memory locations in either the main memory array or in one of the SRAM buffers. The three address bytes will be comprised of a number of dummy bits and a number of actual device address bits, with the number of dummy bits varying depending on the operation being performed and the selected device page size. Buffer addressing for the standard DataFlash page size (528 bytes) is referenced in the datasheet using the terminology BFA9 - BFA0 to denote the 10 address bits required to designate a byte address within a buffer. The main memory addressing is referenced using the terminology PA11 - PA0 and BA9 - BA0, where PA11 - PA0 denotes the 12 address bits required to designate a page address, and BA9 - BA0 denotes the 10 address bits required to designate a byte address within the page. Therefore, when using the standard DataFlash page size, a total of 22 address bits are used.

For the "power of 2" binary page size (512 bytes), the buffer addressing is referenced in the datasheet using the conventional terminology BFA8 - BFA0 to denote the 9 address bits required to designate a byte address within a buffer. Main memory addressing is referenced using the terminology A20 - A0, where A20 - A9 denotes the 12 address bits required to designate a page address, and A8 - A0 denotes the 9 address bits required to designate a byte address within a page. Therefore, when using the binary page size, a total of 21 address bits are used.

4.1   Dual-I/O and Quad I/O Operation

The AT45DQ161 features a Dual-input Buffer Write mode and a Dual-output Read mode that allows two bits of data to be clocked into Buffer 1 or Buffer 2 or allows two bits of data to be read out of the device on every clock cycle to improve throughputs. To accomplish this, both the SI and SO pins are utilized as inputs/outputs for the transfer of data bytes. With the Dual-input Buffer Write command, the SO pin becomes an input along with the SI pin. Alternatively, with the Dual-output Read Array command, the SI pin becomes an output along with the SO pin. For both Dual-I/O commands, the SO pin will be referrred to as I/O_1 and the SI pin will be referred to as I/O_0.

The device also supports a Quad-input Buffer Write mode and a Quad-output Read mode in which the $\overline{WP}$ and $\overline{RESET}$ pins become data pins for even higher throughputs by allowing four bits of data to be clocked on every clock cycle into one of the buffers or by allowing four bits of data to be read out of the device on every clock cycle. For the Quad-input Buffer Write and Quad-output Read Array commands, the $\overline{RESET}$, $\overline{WP}$, SO and SI pins are referred to as I/O_{3-0} where $\overline{RESET}$ becomes I/O_3, $\overline{WP}$ becomes I/O_2, SO becomes I/O_1 and SI becomes I/O_0. The QE bit in the Configuration Register must be set (via issuing the Quad Enable command) to enable the Quad-I/O operation and to enable the $\overline{RESET}$ and $\overline{WP}$ pins to be converted to I/O data pins.

5.   Read Commands

By specifying the appropriate opcode, data can be read from the main memory or from either one of the two SRAM data buffers. The DataFlash supports RapidS protocols for Mode 0 and Mode 3. Please see <blue>Section 25., Detailed Bit-level Read Waveforms: RapidS Mode 0/Mode 3 diagrams in this datasheet for details on the clock cycle sequences for each mode.

5.1   Continuous Array Read (Legacy Command: E8h Opcode)

By supplying an initial starting address for the main memory array, the Continuous Array Read command can be utilized to sequentially read a continuous stream of data from the device by simply providing a clock signal; no additional addressing information or control signals need to be provided. The DataFlash incorporates an internal address counter that will automatically increment on every clock cycle, allowing one continuous read from memory to be performed without the need for additional address sequences. To perform a Continuous Array Read using the standard DataFlash page size (528 bytes), an opcode of E8h must be clocked into the device followed by three address bytes (which comprise the 22-bit page and byte address sequence) and 4 dummy bytes. The first 12 bits (PA11 - PA0) of the 22-bit address sequence specify which page of the main memory array to read and the last 10 bits (BA9 - BA0) of the 22-bit address sequence specify the starting byte address within the page. To perform a Continuous Array Read using the binary page size (512 bytes), an opcode of E8h must be clocked into the device followed by three address bytes and four dummy bytes. The first 12 bits (A20 - A9) of the 21-bit address sequence specify which page of the main memory array to read and the last nine bits (A8 - A0) of the 21-bit address sequence specify the starting byte address within the page. The dummy bytes that follow the address bytes are needed to initialize the read operation. Following the dummy bytes, additional clock pulses on the SCK pin will result in data being output on the SO (serial output) pin.

The $\overline{CS}$ pin must remain low during the loading of the opcode, the address bytes, the dummy bytes, and the reading of data. When the end of a page in the main memory is reached during a Continuous Array Read, the device will continue reading at the beginning of the next page with no delays incurred during the page boundary crossover (the crossover from the end of one page to the beginning of the next page). When the last bit in the main memory array has been read, the device will continue reading back at the beginning of the first page of memory. As with crossing over page boundaries, no delays will be incurred when wrapping around from the end of the array to the beginning of the array.

A low-to-high transition on the $\overline{CS}$ pin will terminate the read operation and tri-state the output pin (SO). The maximum SCK frequency allowable for the Continuous Array Read is defined by the f_{CAR1} specification. The Continuous Array Read bypasses the data buffers and leaves the contents of the buffers unchanged.

5.2   Continuous Array Read (High Frequency Mode: 1Bh Opcode)

This command can be used to read the main memory array sequentially at the highest possible operating clock frequency up to the maximum specified by f_{CAR1}. To perform a Continuous Array Read using the standard DataFlash page size (528 bytes), the $\overline{CS}$ pin must first be asserted, and then an opcode of 1Bh must be clocked into the device followed by three address bytes and two dummy bytes. The first 12 bits (PA11 - PA0) of the 22-bit address sequence specify which page of the main memory array to read and the last 10 bits (BA9 - BA0) of the 22-bit address sequence specify the starting byte address within the page. To perform a Continuous Array Read using the binary page size (512 bytes), the opcode 1Bh must be clocked into the device followed by three address bytes (A20 - A0) and two dummy bytes. Following the dummy bytes, additional clock pulses on the SCK pin will result in data being output on the SO (Serial Output) pin.

The $\overline{CS}$ pin must remain low during the loading of the opcode, the address bytes, the dummy bytes, and the reading of data. When the end of a page in the main memory is reached during a Continuous Array Read, the device will continue reading at the beginning of the next page with no delays incurred during the page boundary crossover (the crossover from the end of one page to the beginning of the next page). When the last bit in the main memory array has been read, the device will continue reading back at the beginning of the first page of memory. As with crossing over page boundaries, no delays will be incurred when wrapping around from the end of the array to the beginning of the array.

A low-to-high transition on the $\overline{CS}$ pin will terminate the read operation and tri-state the output pin (SO). The maximum SCK frequency allowable for the Continuous Array Read is defined by the f_{CAR1} specification. The Continuous Array Read bypasses both data buffers and leaves the contents of the buffers unchanged.

5.3 Continuous Array Read (High Frequency Mode: 0Bh Opcode)

This command can be used to read the main memory array sequentially at higher clock frequencies up to the maximum specified by f_{CAR1}. To perform a Continuous Array Read using the standard DataFlash page size (528 bytes), the $\overline{CS}$ pin must first be asserted, and then an opcode of 0Bh must be clocked into the device followed by three address bytes and one dummy byte. The first 12 bits (PA11 - PA0) of the 22-bit address sequence specify which page of the main memory array to read and the last 10 bits (BA9 - BA0) of the 22-bit address sequence specify the starting byte address within the page. To perform a Continuous Array Read using the binary page size (512 bytes), the opcode 0Bh must be clocked into the device followed by three address bytes (A20 - A0) and one dummy byte. Following the dummy byte, additional clock pulses on the SCK pin will result in data being output on the SO pin.

The $\overline{CS}$ pin must remain low during the loading of the opcode, the address bytes, the dummy byte, and the reading of data. When the end of a page in the main memory is reached during a Continuous Array Read, the device will continue reading at the beginning of the next page with no delays incurred during the page boundary crossover (the crossover from the end of one page to the beginning of the next page). When the last bit in the main memory array has been read, the device will continue reading back at the beginning of the first page of memory. As with crossing over page boundaries, no delays will be incurred when wrapping around from the end of the array to the beginning of the array.

A low-to-high transition on the $\overline{CS}$ pin will terminate the read operation and tri-state the output pin (SO). The maximum SCK frequency allowable for the Continuous Array Read is defined by the f_{CAR1} specification. The Continuous Array Read bypasses both data buffers and leaves the contents of the buffers unchanged.

5.4 Continuous Array Read (Low Frequency Mode: 03h Opcode)

This command can be used to read the main memory array sequentially at lower clock frequencies up to maximum specified by f_{CAR2}. Unlike the previously described read commands, this Continuous Array Read command for the lower clock frequencies does not require the clocking in of dummy bytes after the address byte sequence. To perform a Continuous Array Read using the standard DataFlash page size (528 bytes), the $\overline{CS}$ pin must first be asserted, and then an opcode of 03h must be clocked into the device followed by three address bytes. The first 12 bits (PA11 - PA0) of the 22-bit address sequence specify which page of the main memory array to read and the last 10 bits (BA9 - BA0) of the 22-bit address sequence specify the starting byte address within the page. To perform a Continuous Array Read using the binary page size (512 bytes), the opcode 03h must be clocked into the device followed by three address bytes (A20 - A0). Following the address bytes, additional clock pulses on the SCK pin will result in data being output on the SO pin.

The $\overline{CS}$ pin must remain low during the loading of the opcode, the address bytes, and the reading of data. When the end of a page in the main memory is reached during a Continuous Array Read, the device will continue reading at the beginning of the next page with no delays incurred during the page boundary crossover (the crossover from the end of one page to the beginning of the next page). When the last bit in the main memory array has been read, the device will continue reading back at the beginning of the first page of memory. As with crossing over page boundaries, no delays will be incurred when wrapping around from the end of the array to the beginning of the array.

A low-to-high transition on the $\overline{CS}$ pin will terminate the read operation and tri-state the output pin (SO). The maximum SCK frequency allowable for the Continuous Array Read is defined by the f_{CAR2} specification. The Continuous Array Read bypasses both data buffers and leaves the contents of the buffers unchanged.

5.5 Continuous Array Read (Low Power Mode: 01h Opcode)

This command is ideal for applications that want to minimize power consumption and do not need to read the memory array at high frequencies. Like the 03h opcode, this Continuous Array Read command allows reading the main memory array sequentially without the need for dummy bytes to be clocked in after the address byte sequence. The memory can be read at clock frequencies up to maximum specified by f_{CAR3}. To perform a Continuous Array Read using the standard DataFlash page size (528 bytes), the $\overline{CS}$ pin must first be asserted, and then an opcode of 01h must be clocked into the device followed by three address bytes. The first 12 bits (PA11 - PA0) of the 22-bit address sequence specify which page of the main memory array to read and the last 10 bits (BA9 - BA0) of the 22-bit address sequence specify the starting byte address within the page. To perform a Continuous Array Read using the binary page size (512 bytes), the opcode 01h must be clocked into the device followed by three address bytes (A20 - A0). Following the address bytes, additional clock pulses on the SCK pin will result in data being output on the SO pin.

The $\overline{CS}$ pin must remain low during the loading of the opcode, the address bytes, and the reading of data. When the end of a page in the main memory is reached during a Continuous Array Read, the device will continue reading at the beginning of the next page with no delays incurred during the page boundary crossover (the crossover from the end of one page to the beginning of the next page). When the last bit in the main memory array has been read, the device will continue reading back at the beginning of the first page of memory. As with crossing over page boundaries, no delays will be incurred when wrapping around from the end of the array to the beginning of the array.

A low-to-high transition on the $\overline{CS}$ pin will terminate the read operation and tri-state the output pin (SO). The maximum SCK frequency allowable for the Continuous Array Read is defined by the f_{CAR3} specification. The Continuous Array Read bypasses both data buffers and leaves the contents of the buffers unchanged.

5.6   Main Memory Page Read

A Main Memory Page Read allows the reading of data directly from a single page in the main memory, bypassing both of the data buffers and leaving the contents of the buffers unchanged. To start a page read using the standard DataFlash page size (528 bytes), an opcode of D2h must be clocked into the device followed by three address bytes (which comprise the 22-bit page and byte address sequence) and 4 dummy bytes. The first 12 bits (PA11 - PA0) of the 22-bit address sequence specify the page in main memory to be read and the last 10 bits (BA9 - BA0) of the 22-bit address sequence specify the starting byte address within that page. To start a page read using the binary page size (512 bytes), the opcode D2h must be clocked into the device followed by three address bytes and four dummy bytes. The first 12 bits (A20 - A9) of the 21-bit address sequence specify which page of the main memory array to read, and the last nine bits (A8 - A0) of the 21-bit address sequence specify the starting byte address within that page. The dummy bytes that follow the address bytes are sent to initialize the read operation. Following the dummy bytes, the additional pulses on SCK result in data being output on the SO (serial output) pin.

The $\overline{CS}$ pin must remain low during the loading of the opcode, the address bytes, the dummy bytes, and the reading of data. Unlike the Continuous Array Read command, when the end of a page in main memory is reached, the device will continue reading back at the beginning of the same page rather than the beginning of the next page.

A low-to-high transition on the $\overline{CS}$ pin will terminate the read operation and tri-state the output pin (SO). The maximum SCK frequency allowable for the Main Memory Page Read is defined by the f_{SCK} specification. The Main Memory Page Read bypasses both data buffers and leaves the contents of the buffers unchanged.

5.7   Buffer Read

The SRAM data buffers can be accessed independently from the main memory array, and utilizing the Buffer Read command allows data to be sequentially read directly from either one of the buffers. Four opcodes, D4h or D1h for Buffer 1 and D6h or D3h for Buffer 2, can be used for the Buffer Read command. The use of each opcode depends on the maximum SCK frequency that will be used to read data from the buffers. The D4h and D6h opcode can be used at any SCK frequency up to the maximum specified by f_{MAX} while the D1h and D3h opcode can be used for lower frequency read operations up to the maximum specified by f_{CAR2}.

To perform a Buffer Read using the standard DataFlash buffer size (528 bytes), the opcode must be clocked into the device followed by three address bytes comprised of 14 dummy bits and 10 buffer address bits (BFA9 - BFA0). To perform a Buffer Read using the binary buffer size (512 bytes), the opcode must be clocked into the device followed by three address bytes comprised of 15 dummy bits and 9 buffer address bits (BFA8 - BFA0). Following the address bytes, one dummy byte must be clocked into the device to initialize the read operation if using opcodes D4h or D6h. The $\overline{CS}$ pin must remain low during the loading of the opcode, the address bytes, the dummy byte (if using opcodes D4h or D6h), and the reading of data. When the end of a buffer is reached, the device will continue reading back at the beginning of the buffer. A low-to-high transition on the $\overline{CS}$ pin will terminate the read operation and tri-state the output pin (SO).

5.8 Dual-output Read Array

The Dual-output Read Array command is similar to the Continuous Array Read command and can be used to sequentially read a continuous stream of data from the device by simply providing the clock signal once the initial starting address has been specified. Unlike the Continuous Array Read command however, the Dual-output Read Array command allows two bits of data to be clocked out of the device on every clock cycle rather than just one.

The Dual-output Read Array command can be used at any clock frequency up to the maximum specified by f_{SCK}. To perform a Dual-output Read Array using the standard DataFlash page size (528 bytes), the $\overline{CS}$ pin must first be asserted, and then an opcode of 3Bh must be clocked into the device followed by three address bytes and one dummy byte. The first 12 bits (PA11 - PA0) of the 22-bit address sequence specify which page of the main memory array to read and the last 10 bits (BA9 - BA0) of the 22-bit address sequence specify the starting byte address within the page.

To perform a Dual-output Read Array using the binary page size (512 bytes), the opcode 3Bh must be clocked into the device followed by three address bytes (A20 - A0) and one dummy byte.

After the three address bytes and the dummy byte have been clocked in, additional clock cycles will result in data being output on both the I/O_1 and I/O_0 pins. The data is always output with the MSB of a byte first, and the MSB is always output on the I/O_1 pin. During the first clock cycle, bit seven of the first data byte will be output on the I/O_1 pin while bit six of the same data byte will be output on the I/O_0 pin. During the next clock cycle, bits five and four of the first data byte will be output on the I/O_1 and I/O_0 pins, respectively. The sequence continues with each byte of data being output after every four clock cycles.

The $\overline{CS}$ pin must remain low during the loading of the opcode, the address bytes, the dummy byte, and the reading of data. When the end of a page in the main memory is reached during a Dual-output Read Array the device will continue reading at the beginning of the next page with no delays incurred during the page boundary crossover (the crossover from the end of one page to the beginning of the next page). When the last bit in the main memory array has been read, the device will continue reading back at the beginning of the first page of memory. As with crossing over page boundaries, no delays will be incurred when wrapping around from the end of the array to the beginning of the array.

A low-to-high transition on the $\overline{CS}$ pin will terminate the read operation and tri-state both the I/O_1 and I/O_0 pins. The Dual-output Dual-output Read Array bypasses both data buffers and leaves the contents of the buffers unchanged.

5.9 Quad-output Read Array

The Quad-output Read Array command is similar to the Dual-output Read Array command and can be used to sequentially read a continuous stream of data from the device by simply providing the clock signal once the initial starting address has been specified. Unlike the Dual-output Read Array command however, the Quad-output Read Array command allows four bits of data to be clocked out of the device on every clock cycle rather than two.

> Note: The QE bit in the Configuration Register must be previously set in order for any Quad-I/O command (i.e. Quad-output Read Array command) to be enabled and for the $\overline{RESET}$ and $\overline{WP}$ pins to be converted to I/O data pins.

The Quad-output Read Array command can be used at any clock frequency up to the maximum specified by f_{SCK}. To perform a Quad-output Read Array using the standard DataFlash page size (528 bytes), the $\overline{CS}$ pin must first be asserted, and then an opcode of 6Bh must be clocked into the device followed by three address bytes and one dummy byte. The first 12 bits (PA11 - PA0) of the 22-bit address sequence specify which page of the main memory array to read and the last 10 bits (BA9 - BA0) of the 22-bit address sequence specify the starting byte address within the page.

To perform a Quad-output Read Array using the binary page size (512 bytes), the opcode 6Bh must be clocked into the device followed by three address bytes (A20 - A0) and one dummy byte.

After the three address bytes and the dummy byte have been clocked in, additional clock cycles will result in data being output on the I/O_{3-0} pins. The data is always output with the MSB of a byte first and the MSB is always output on the I/O_3 pin. During the first clock cycle, bit seven of the first data byte will be output on the I/O_3 pin while bits six, five, and four of the same data byte will be output on the I/O_2, I/O_1, and I/O_0 pins, respectively. During the next clock cycle, bits three, two, one, and zero of the first data byte will be output on the I/O_3, I/O_2, I/O_1 and I/O_0 pins, respectively. The sequence continues with each byte of data being output after every two clock cycles.

The $\overline{CS}$ pin must remain low during the loading of the opcode, the address bytes, the dummy byte, and the reading of data. When the end of a page in the main memory is reached during a Quad-output Read Array the device will continue reading at the beginning of the next page with no delays incurred during the page boundary crossover (the crossover from the end of one page to the beginning of the next page). When the last bit in the main memory array has been read, the device will continue reading back at the beginning of the first page of memory. As with crossing over page boundaries, no delays will be incurred when wrapping around from the end of the array to the beginning of the array.

A low-to-high transition on the $\overline{CS}$ pin will terminate the read operation and tri-state the I/O$_3$, I/O$_2$, I/O$_1$ and I/O$_0$ pins. The Quad-output Read Array bypasses both data buffers and leaves the contents of the buffers unchanged.

6. Program and Erase Commands

6.1 Buffer Write

Utilizing the Buffer Write command allows data clocked in from the SI pin to be written directly into either one of the SRAM data buffers.

To load data into a buffer using the standard DataFlash buffer size (528 bytes), an opcode of 84h for Buffer 1 or 87h for Buffer 2 must be clocked into the device followed by three address bytes comprised of 14 dummy bits and 10 buffer address bits (BFA9 - BFA0). The 10 buffer address bits specify the first byte in the buffer to be written.

To load data into a buffer using the binary buffer size (512 bytes), an opcode of 84h for Buffer 1 or 87h for Buffer 2, must be clocked into the device followed by 15 dummy bits and 9 buffer address bits (BFA8 - BFA0). The 9 buffer address bits specify the first byte in the buffer to be written.

After the last address byte has been clocked into the device, data can then be clocked in on subsequent clock cycles. If the end of the data buffer is reached, the device will wrap around back to the beginning of the buffer. Data will continue to be loaded into the buffer until a low-to-high transition is detected on the $\overline{CS}$ pin.

6.2 Dual-input Buffer Write

The Dual-input Buffer Write command is similar to the Buffer Write command and can be used to increase the data input into one of the SRAM buffers by allowing two bits of data to be clocked into the device on every clock cycle rather than just one.

To load data into a buffer using the standard DataFlash buffer size (528 bytes), an opcode of 24h for Buffer 1 or 27h for Buffer 2 must be clocked into the device followed by three address bytes comprised of 14 dummy bits and 10 buffer address bits (BFA9 - BFA0). The 10 buffer address bits specify the first byte in the buffer to be written.

To load data into a buffer using the binary buffer size (512 bytes), an opcode of 24h for Buffer 1 or 27h for Buffer 2, must be clocked into the device followed by 15 dummy bits and 9 buffer address bits (BFA8 - BFA0). The 9 buffer address bits specify the first byte in the buffer to be written.

After the last address byte has been clocked into the device, data can then be clocked in on subsequent clock cycles. If the end of the data buffer is reached, the device will wrap around back to the beginning of the buffer. Data will continue to be loaded into the buffer until a low-to-high transition is detected on the $\overline{CS}$ pin.

6.3 Quad-input Buffer Write

The Quad-input Buffer Write command is similar to the Buffer Write command and can be used to significantly increase the data input into one of the SRAM buffers by allowing four bits of data to be clocked into the device on every clock cycle rather than just one.

To load data into a buffer using the standard DataFlash buffer size (528 bytes), an opcode of 44h for Buffer 1 or 47h for Buffer 2 must be clocked into the device followed by three address bytes comprised of 14 dummy bits and 10 buffer address bits (BFA9 - BFA0). The 10 buffer address bits specify the first byte in the buffer to be written.

To load data into a buffer using the binary buffer size (512 bytes), an opcode of 44h for Buffer 1 or 47h for Buffer 2, must be clocked into the device followed by 15 dummy bits and 9 buffer address bits (BFA8 - BFA0). The 9 buffer address bits specify the first byte in the buffer to be written.

After the last address byte has been clocked into the device, data can then be clocked in on subsequent clock cycles. If the end of the data buffer is reached, the device will wrap around back to the beginning of the buffer. Data will continue to be loaded into the buffer until a low-to-high transition is detected on the $\overline{CS}$ pin.

6.4   Buffer to Main Memory Page Program with Built-In Erase

The Buffer to Main Memory Page Program with Built-In Erase command allows data that is stored in one of the SRAM buffers to be written into an erased or programmed page in the main memory array. It is not necessary to pre-erase the page in main memory to be written because this command will automatically erase the selected page prior to the program cycle.

To perform a Buffer to Main Memory Page Program with Built-In Erase using the standard DataFlash page size (528 bytes), an opcode of 83h for Buffer 1 or 86h for Buffer 2 must be clocked into the device followed by three address bytes comprised of 2 dummy bits, 12 page address bits (PA11 - PA0) that specify the page in the main memory to be written, and 10 dummy bits.

To perform a Buffer to Main Memory Page Program with Built-In Erase using the binary page size (512 bytes), an opcode of 83h for Buffer 1 or 86h for Buffer 2 must be clocked into the device followed by three address bytes comprised of 3 dummy bits, 12 page address bits (A20 - A9) that specify the page in the main memory to be written, and 9 dummy bits.

When a low-to-high transition occurs on the $\overline{CS}$ pin, the device will first erase the selected page in main memory (the erased state is a Logic 1) and then program the data stored in the appropriate buffer into that same page in main memory. Both the erasing and the programming of the page are internally self-timed and should take place in a maximum time of t_{EP}. During this time, the RDY/$\overline{BUSY}$ bit in the Status Register will indicate that the device is busy.

The device also incorporates an intelligent erase and program algorithm that can detect when a byte location fails to erase or program properly. If an erase or programming error arises, it will be indicated by the EPE bit in the Status Register.

6.5   Buffer to Main Memory Page Program without Built-In Erase

The Buffer to Main Memory Page Program without Built-In Erase command allows data that is stored in one of the SRAM buffers to be written into a pre-erased page in the main memory array. It is necessary that the page in main memory to be written be previously erased in order to avoid programming errors.

To perform a Buffer to Main Memory Page Program without Built-In Erase using the standard DataFlash page size (528 bytes), an opcode of 88h for Buffer 1 or 89h for Buffer 2 must be clocked into the device followed by three address bytes comprised of 2 dummy bits, 12 page address bits (PA11 - PA0) that specify the page in the main memory to be written, and 10 dummy bits.

To perform a Buffer to Main Memory Page Program using the binary page size (512 bytes), an opcode of 88h for Buffer 1 or 89h for Buffer 2 must be clocked into the device followed by three address bytes comprised of three dummy bits, 12 page address bits (A20 - A9) that specify the page in the main memory to be written, and 9 dummy bits.

When a low-to-high transition occurs on the $\overline{CS}$ pin, the device will program the data stored in the appropriate buffer into the specified page in the main memory. The page in main memory that is being programmed must have been previously erased using one of the erase commands (Page Erase, Block Erase, Sector Erase, or Chip Erase). The programming of the page is internally self-timed and should take place in a maximum time of t_P. During this time, the RDY/$\overline{BUSY}$ bit in the Status Register will indicate that the device is busy.

The device also incorporates an intelligent programming algorithm that can detect when a byte location fails to program properly. If a programming error arises, it will be indicated by the EPE bit in the Status Register.

6.6   Main Memory Page Program through Buffer with Built-In Erase

The Main Memory Page Program through Buffer with Built-In Erase command combines the Buffer Write and Buffer to Main Memory Page Program with Built-In Erase operations into a single operation to help simplify application firmware development. With the Main Memory Page Program through Buffer with Built-In Erase command, data is first clocked into either Buffer 1 or Buffer 2, the addressed page in memory is then automatically erased, and then the contents of the appropriate buffer are programmed into the just-erased main memory page.

To perform a Main Memory Page Program through Buffer using the standard DataFlash page size (528 bytes), an opcode of 82h for Buffer 1 or 85h for Buffer 2 must first be clocked into the device followed by three address bytes comprised of 2 dummy bits, 12 page address bits (PA11 - PA0) that specify the page in the main memory to be written, and 10 buffer address bits (BFA9 - BFA0) that select the first byte in the buffer to be written.

To perform a Main Memory Page Program through Buffer using the binary page size (512 bytes), an opcode of 82h for Buffer 1 or 85h for Buffer 2 must first be clocked into the device followed by three address bytes comprised of three dummy bits, 12 page address bits (A20 - A9) that specify the page in the main memory to be written, and 9 buffer address bits (BFA8 - BFA0) that select the first byte in the buffer to be written.

After all address bytes have been clocked in, the device will take data from the input pin (SI) and store it in the specified data buffer. If the end of the buffer is reached, the device will wrap around back to the beginning of the buffer. When there is a low-to-high transition on the $\overline{CS}$ pin, the device will first erase the selected page in main memory (the erased state is a Logic 1) and then program the data stored in the buffer into that main memory page. Both the erasing and the programming of the page are internally self-timed and should take place in a maximum time of t_{EP}. During this time, the RDY/$\overline{BUSY}$ bit in the Status Register will indicate that the device is busy.

The device also incorporates an intelligent erase and programming algorithm that can detect when a byte location fails to erase or program properly. If an erase or program error arises, it will be indicated by the EPE bit in the Status Register.

6.7   Main Memory Byte/Page Program through Buffer 1 without Built-In Erase

The Main Memory Byte/Page Program through Buffer 1 without Built-In Erase command combines both the Buffer Write and Buffer to Main Memory Program without Built-In Erase operations to allow any number of bytes (1 to 512/528 bytes) to be programmed directly into previously erased locations in the main memory array. With the Main Memory Byte/Page Program through Buffer 1 without Built-In Erase command, data is first clocked into Buffer 1, and then only the bytes clocked into the buffer are programmed into the pre-erased byte locations in main memory. Multiple bytes up to the page size can be entered with one command sequence.

To perform a Main Memory Byte/Page Program through Buffer 1 using the standard DataFlash page size (528 bytes), an opcode of 02h must first be clocked into the device followed by three address bytes comprised of 2 dummy bits, 12 page address bits (PA11 - PA0) that specify the page in the main memory to be written, and 10 buffer address bits (BFA9 - BFA0) that select the first byte in the buffer to be written. After all address bytes are clocked in, the device will take data from the input pin (SI) and store it in Buffer 1. Any number of bytes (1 to 528) can be entered. If the end of the buffer is reached, then the device will wrap around back to the beginning of the buffer.

To perform a Main Memory Byte/Page Program through Buffer 1 using the binary page size (512 bytes), an opcode of 02h for Buffer 1 using must first be clocked into the device followed by three address bytes comprised of three dummy bits, 12 page address bits (A20 - A9) that specify the page in the main memory to be written, and 9 buffer address bits (BFA8 - BFA0) that selects the first byte in the buffer to be written. After all address bytes are clocked in, the device will take data from the input pin (SI) and store it in Buffer 1. Any number of bytes (1 to 512) can be entered. If the end of the buffer is reached, then the device will wrap around back to the beginning of the buffer. When using the binary page size, the page and buffer address bits correspond to a 21-bit logical address (A20-A0) in the main memory.

After all data bytes have been clocked into the device, a low-to-high transition on the $\overline{CS}$ pin will start the program operation in which the device will program the data stored in Buffer 1 into the main memory array. Only the data bytes that were clocked into the device will be programmed into the main memory.

Example: If only two data bytes were clocked into the device, then only two bytes will be programmed into main memory and the remaining bytes in the memory page will remain in their previous state.

The $\overline{CS}$ pin must be deasserted on a byte boundary (multiples of eight bits); otherwise, the operation will be aborted and no data will be programmed. The programming of the data bytes is internally self-timed and should take place in a maximum time of t_P (the program time will be a multiple of the t_{BP} time depending on the number of bytes being programmed). During this time, the RDY/$\overline{BUSY}$ bit in the Status Register will indicate that the device is busy.

The device also incorporates an intelligent programming algorithm that can detect when a byte location fails to program properly. If a programming error arises, it will be indicated by the EPE bit in the Status Register.

6.8   Page Erase

The Page Erase command can be used to individually erase any page in the main memory array allowing the Buffer to Main Memory Page Program without Built-In Erase command or the Main Memory Byte/Page Program through Buffer 1 command to be utilized at a later time.

To perform a Page Erase with the standard DataFlash page size (528 bytes), an opcode of 81h must be clocked into the device followed by three address bytes comprised of 2 dummy bits, 12 page address bits (PA11 - PA0) that specify the page in the main memory to be erased, and 10 dummy bits.

To perform a Page Erase with the binary page size (512 bytes), an opcode of 81h must be clocked into the device followed by three address bytes comprised of three dummy bits, 12 page address bits (A20 - A9) that specify the page in the main memory to be erased, and 9 dummy bits.

When a low-to-high transition occurs on the $\overline{CS}$ pin, the device will erase the selected page (the erased state is a Logic 1). The erase operation is internally self-timed and should take place in a maximum time of t_{PE}. During this time, the RDY/$\overline{BUSY}$ bit in the Status Register will indicate that the device is busy.

The device also incorporates an intelligent erase algorithm that can detect when a byte location fails to erase properly. If an erase error arises, it will be indicated by the EPE bit in the Status Register.

6.9   Block Erase

The Block Erase command can be used to erase a block of eight pages at one time. This command is useful when needing to pre-erase larger amounts of memory and is more efficient than issuing eight separate Page Erase commands.

To perform a Block Erase with the standard DataFlash page size (528 bytes), an opcode of 50h must be clocked into the device followed by three address bytes comprised of 2 dummy bits, 9 page address bits (PA11 - PA3), and 13 dummy bits. The 9 page address bits are used to specify which block of eight pages is to be erased.

To perform a Block Erase with the binary page size (512 bytes), an opcode of 50h must be clocked into the device followed by three address bytes comprised of three dummy bits, 9 page address bits (A20 - A12), and 12 dummy bits. The 9 page address bits are used to specify which block of eight pages is to be erased.

When a low-to-high transition occurs on the $\overline{CS}$ pin, the device will erase the selected block of eight pages. The erase operation is internally self-timed and should take place in a maximum time of t_{BE}. During this time, the RDY/$\overline{BUSY}$ bit in the Status Register will indicate that the device is busy.

The device also incorporates an intelligent erase algorithm that can detect when a byte location fails to erase properly. If an erase error arises, it will be indicated by the EPE bit in the Status Register.

Table 6-1.   Block Erase Addressing

PA11/ A20	PA10/ A19	PA9/ A18	PA8/ A17	PA7/ A16	PA6/ A15	PA5/ A14	PA4/ A13	PA3/ A12	PA2/ A11	PA1/ A10	PA0/ A9	Block
0	0	0	0	0	0	0	0	0	X	X	X	0
0	0	0	0	0	0	0	0	1	X	X	X	1
0	0	0	0	0	0	0	1	0	X	X	X	2
0	0	0	0	0	0	0	1	1	X	X	X	3
•	•	•	•	•	•	•	•	•	•	•	•	•
•	•	•	•	•	•	•	•	•	•	•	•	•
•	•	•	•	•	•	•	•	•	•	•	•	•
1	1	1	1	1	1	1	0	0	X	X	X	508
1	1	1	1	1	1	1	0	1	X	X	X	509
1	1	1	1	1	1	1	1	0	X	X	X	510
1	1	1	1	1	1	1	1	1	X	X	X	511

6.10   Sector Erase

The Sector Erase command can be used to individually erase any sector in the main memory.

The main memory array is comprised of 17 sectors, and only one sector can be erased at a time. To perform an erase of Sector 0a or Sector 0b with the standard DataFlash page size (528 bytes), an opcode of 7Ch must be clocked into the device followed by three address bytes comprised of 2 dummy bits, 9 page address bits (PA11 - PA3), and 13 dummy bits. To perform a Sector 1-15 erase, an opcode of 7Ch must be clocked into the device followed by three address bytes comprised of 2 dummy bits, 4 page address bits (PA11 - PA8), and 18 dummy bits.

To perform a Sector 0a or Sector 0b erase with the binary page size (512 bytes), an opcode of 7Ch must be clocked into the device followed by three address bytes comprised of three dummy bits, 9 page address bits (A20 - A12), and 12 dummy bits. To perform a Sector 1-15 erase, an opcode of 7Ch must be clocked into the device followed by 3 dummy bits, 4 page address bits (A20 - A17), and 17 dummy bits.

The page address bits are used to specify any valid address location within the sector to be erased. When a low-to high transition occurs on the $\overline{CS}$ pin, the device will erase the selected sector. The erase operation is internally self-timed and should take place in a maximum time of t_{SE}. During this time, the RDY/$\overline{BUSY}$ bit in the Status Register will indicate that the device is busy.

The device also incorporates an intelligent algorithm that can detect when a byte location fails to erase properly. If an erase error arises, it will be indicated by the EPE bit in the Status Register.

Table 6-2. Sector Erase Addressing

PA11/ A20	PA10/ A19	PA9/ A18	PA8/ A17	PA7/ A16	PA6/ A15	PA5/ A14	PA4/ A13	PA3/ A12	PA2/ A11	PA1/ A10	PA0/ A9	Sector
0	0	0	0	0	0	0	0	0	X	X	X	0a
0	0	0	0	0	0	0	0	1	X	X	X	0b
0	0	0	1	X	X	X	X	X	X	X	X	1
0	0	1	0	X	X	X	X	X	X	X	X	2
•	•	•	•	•	•	•	•	•	•	•	•	•
•	•	•	•	•	•	•	•	•	•	•	•	•
•	•	•	•	•	•	•	•	•	•	•	•	•
1	1	0	0	X	X	X	X	X	X	X	X	12
1	1	0	1	X	X	X	X	X	X	X	X	13
1	1	1	0	X	X	X	X	X	X	X	X	14
1	1	1	1	X	X	X	X	X	X	X	X	15

6.11 Chip Erase

The Chip Erase command allows the entire main memory array to be erased can be erased at one time.

To execute the Chip Erase command, a 4-byte command sequence of C7h, 94h, 80h, and 9Ah must be clocked into the device. Since the entire memory array is to be erased, no address bytes need to be clocked into the device, and any data clocked in after the opcode will be ignored. After the last bit of the opcode sequence has been clocked in, the $\overline{CS}$ pin must be deasserted to start the erase process. The erase operation is internally self-timed and should take place in a time of t_{CE}. During this time, the RDY/$\overline{BUSY}$ bit in the Status Register will indicate that the device is busy.

The Chip Erase command will not affect sectors that are protected or locked down; the contents of those sectors will remain unchanged. Only those sectors that are not protected or locked down will be erased.

The $\overline{WP}$ pin can be asserted while the device is erasing, but protection will not be activated until the internal erase cycle completes.

The device also incorporates an intelligent algorithm that can detect when a byte location fails to erase properly. If an erase error arises, it will be indicated by the EPE bit in the Status Register.

Table 6-3. Chip Erase Command

Command	Byte 1	Byte 2	Byte 3	Byte 4
Chip Erase	C7h	94h	80h	9Ah

Figure 6-1. Chip Erase

Each transition represents eight bits
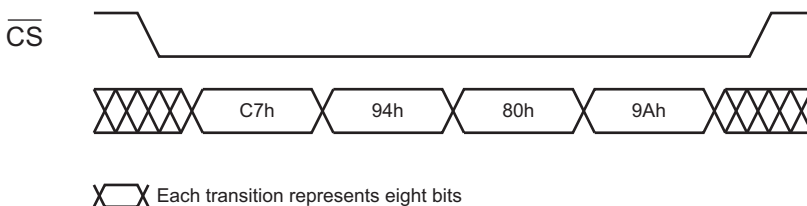
6.12 Program/Erase Suspend

In some code and data storage applications, it may not be possible for the system to wait the milliseconds required for the Flash memory to complete a program or erase cycle. The Program/Erase Suspend command allows a program or erase operation in progress to a particular 128KB sector of the main memory array to be suspended so that other device operations can be performed.

Example: By suspending an erase operation to a particular sector, the system can perform functions such as a program or read operation within a different 128KB sector. Other device operations, such as Read Status Register, can also be performed while a program or erase operation is suspended.

To perform a Program/Erase Suspend, an opcode of B0h must be clocked into the device. No address bytes need to be clocked into the device, and any data clocked in after the opcode will be ignored. When the $\overline{CS}$ pin is deasserted, the program or erase operation currently in progress will be suspended within a time of t_{SUSP}. One of the Program Suspend bits (PS1 or PS2) or the Erase Suspend bit (ES) in the Status Register will then be set to the Logic 1 state. In addition, the RDY/$\overline{BUSY}$ bit in the Status Register will indicate that the device is ready for another operation.

Read operations are not allowed to a 128KB sector that has had its program or erase operation suspended. If a read is attempted to a suspended sector, then the device will output undefined data. Therefore, when performing a Continuous Array Read operation and the device's internal address counter increments and crosses the sector boundary to a suspended sector, the device will then start outputting undefined data continuously until the address counter increments and crosses a sector boundary to an unsuspended sector.

A program operation is not allowed to a sector that has been erase suspended. If a program operation is attempted to an erase suspended sector, then the program operation will abort.

During an Erase Suspend, a program operation to a different 128KB sector can be started and subsequently suspended. This results in a simultaneous Erase Suspend/Program Suspend condition and will be indicated by the states of both the ES and PS1 or PS2 bits in the Status Register being set to a Logic 1.

If a Reset command is performed, or if the $\overline{RESET}$ pin is asserted while a sector is erase suspended, then the suspend operation will be aborted and the contents of the sector will be left in an undefined state. However, if a reset is performed while a page is program or erase suspended, the suspend operation will abort but only the contents of the page that was being programmed or erased will be undefined; the remaining pages in the 128KB sector will retain their previous contents.

Table 6-4.   Operations Allowed and Not Allowed During Suspend

Command	Operation During Program Suspend in Buffer 1 (PS1)	Operation During Program Suspend in Buffer 2 (PS2)	Operation During Erase Suspend (ES)
Read Commands			
Read Array (All Opcodes)	Allowed	Allowed	Allowed
Read Buffer 1 (All Opcodes)	Allowed	Allowed	Allowed
Read Buffer 2 (All Opcodes)	Allowed	Allowed	Allowed
Dual-output Read Array	Allowed	Allowed	Allowed
Quad-output Read Array	Allowed	Allowed	Allowed
Read Configuration Register	Allowed	Allowed	Allowed
Read Status Register	Allowed	Allowed	Allowed
Read Manufacturer and Device ID	Allowed	Allowed	Allowed
Program and Erase Commands			
Buffer 1 Write	Not Allowed	Allowed	Allowed
Buffer 2 Write	Allowed	Not Allowed	Allowed
Dual-input Buffer 1 Write	Not Allowed	Allowed	Allowed
Dual-input Buffer 2 Write	Allowed	Not Allowed	Allowed
Quad-input Buffer 1 Write	Not Allowed	Allowed	Allowed
Quad-input Buffer 2 Write	Allowed	Not Allowed	Allowed
Buffer 1 to Memory Program w/ Erase	Not Allowed	Not Allowed	Not Allowed
Buffer 2 to Memory Program w/ Erase	Not Allowed	Not Allowed	Not Allowed
Buffer 1 to Memory Program w/o Erase	Not Allowed	Not Allowed	Allowed
Buffer 2 to Memory Program w/o Erase	Not Allowed	Not Allowed	Allowed
Memory Program through Buffer 1 w/ Erase	Not Allowed	Not Allowed	Not Allowed
Memory Program through Buffer 2 w/ Erase	Not Allowed	Not Allowed	Not Allowed
Memory Program through Buffer 1 w/o Erase	Not Allowed	Not Allowed	Allowed
Auto Page Rewrite	Not Allowed	Not Allowed	Not Allowed
Page Erase	Not Allowed	Not Allowed	Not Allowed
Block Erase	Not Allowed	Not Allowed	Not Allowed
Sector Erase	Not Allowed	Not Allowed	Not Allowed
Chip Erase	Not Allowed	Not Allowed	Not Allowed
Protection and Security Commands			
Enable Sector Protection	Not Allowed	Not Allowed	Not Allowed
Disable Sector Protection	Not Allowed	Not Allowed	Not Allowed
Erase Sector Protection Register	Not Allowed	Not Allowed	Not Allowed
Program Sector Protection Register	Not Allowed	Not Allowed	Not Allowed
Read Sector Protection Register	Allowed	Allowed	Allowed
Sector Lockdown	Not Allowed	Not Allowed	Not Allowed
Read Sector Lockdown	Allowed	Allowed	Allowed
Freeze Sector Lockdown State	Not Allowed	Not Allowed	Not Allowed
Program Security Register	Not Allowed	Not Allowed	Not Allowed
Read Security Register	Allowed	Allowed	Allowed
Additional Commands			
Main Memory to Buffer 1 Transfer	Not Allowed	Allowed	Allowed
Main Memory to Buffer 2 Transfer	Allowed	Not Allowed	Allowed
Main Memory to Buffer 1 Compare	Allowed	Allowed	Allowed
Main Memory to Buffer 2 Compare	Allowed	Allowed	Allowed
Enter Deep Power-Down	Not Allowed	Not Allowed	Not Allowed
Resume from Deep Power-Down	Not Allowed	Not Allowed	Not Allowed
Enter Ultra-Deep Power-Down mode	Not Allowed	Not Allowed	Not Allowed
Reset (via Hardware or Software)	Allowed	Allowed	Allowed

adesto
TECHNOLOGIES

6.13   Program/Erase Resume

The Program/Erase Resume command allows a suspended program or erase operation to be resumed and continue where it left off.

To perform a Program/Erase Resume, an opcode of D0h must be clocked into the device. No address bytes need to be clocked into the device, and any data clocked in after the opcode will be ignored. When the $\overline{CS}$ pin is deasserted, the program or erase operation currently suspended will be resumed within a time of t_{RES}. The PS1 bit, PS2 bit, or ES bit in the Status Register will then be reset back to a Logic 0 state to indicate that the program or erase operation is no longer suspended. In addition, the RDY/$\overline{BUSY}$ bit in the Status Register will indicate that the device is busy performing a program or erase operation.

During a simultaneous Erase Suspend/Program Suspend condition, issuing the Program/Erase Resume command will result in the program operation resuming first. After the program operation has been completed, the Program/Erase Resume command must be issued again in order for the erase operation to be resumed.

While the device is busy resuming a program or erase operation, any attempts at issuing the Program/Erase Suspend command will be ignored. Therefore, if a resumed program or erase operation needs to be subsequently suspended again, the system must either wait the entire t_{RES} time before issuing the Program/Erase Suspend command, or it must check the status of the RDY/$\overline{BUSY}$ bit or the appropriate PS1, PS2, or ES bit in the Status Register to determine if the previously suspended program or erase operation has resumed.

7.   Sector Protection

Two protection methods, hardware and software controlled, are provided for protection against inadvertent or erroneous program and erase cycles. The software controlled method relies on the use of software commands to enable and disable sector protection while the hardware controlled method employs the use of the Write Protect ($\overline{WP}$) pin. The selection of which sectors that are to be protected or unprotected against program and erase operations is specified in the Nonvolatile Sector Protection Register. The status of whether or not sector protection has been enabled or disabled by either the software or the hardware controlled methods can be determined by checking the Status Register.

7.1   Software Sector Protection

Software controlled protection is useful in applications in which the $\overline{WP}$ pin is not or cannot be controlled by a host processor. In such instances, the $\overline{WP}$ pin may be left floating (the $\overline{WP}$ pin is internally pulled high) and sector protection can be controlled using the Enable Sector Protection and Disable Sector Protection commands.

If the device is power cycled, then the software controlled protection will be disabled. Once the device is powered up, the Enable Sector Protection command should be reissued if sector protection is desired and if the $\overline{WP}$ pin is not used.
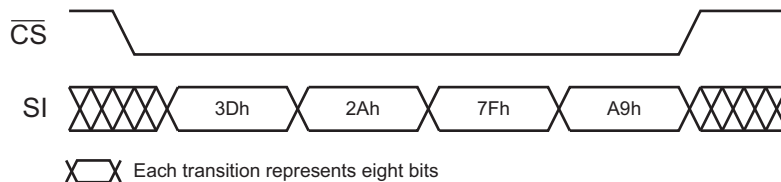
7.1.1   Enable Sector Protection

Sectors specified for protection in the Sector Protection Register can be protected from program and erase operations by issuing the Enable Sector Protection command. To enable the sector protection, a 4-byte command sequence of 3Dh, 2Ah, 7Fh, and A9h must be clocked into the device. After the last bit of the opcode sequence has been clocked in, the $\overline{CS}$ pin must be deasserted to enable the Sector Protection.

Table 7-1.   Enable Sector Protection Command

| Command	Byte 1	Byte 2	Byte 3	Byte 4
Enable Sector Protection	3Dh	2Ah	7Fh	A9h

Figure 7-1.   Enable Sector Protection

7.1.2 Disable Sector Protection

To disable the sector protection, a 4-byte command sequence of 3Dh, 2Ah, 7Fh, and 9Ah must be clocked into the device. After the last bit of the opcode sequence has been clocked in, the $\overline{CS}$ pin must be deasserted to disable the sector protection.

Table 7-2.   Disable Sector Protection Command

Command	Byte 1	Byte 2	Byte 3	Byte 4
Disable Sector Protection	3Dh	2Ah	7Fh	9Ah

Figure 7-2.   Disable Sector Protection

X___X Each transition represents eight bits
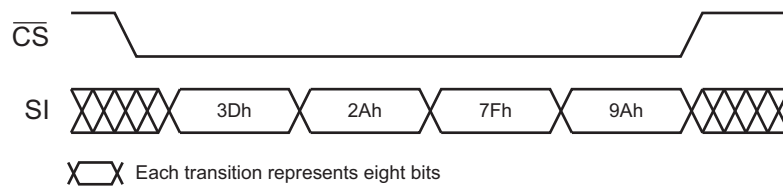
7.2   Hardware Controlled Protection

Sectors specified for protection in the Sector Protection Register and the Sector Protection Register itself can be protected from program and erase operations by asserting the $\overline{WP}$ pin and keeping the pin in its asserted state. The Sector Protection Register and any sector specified for protection cannot be erased or programmed as long as the $\overline{WP}$ pin is asserted. In order to modify the Sector Protection Register, the $\overline{WP}$ pin must be deasserted. If the $\overline{WP}$ pin is permanently connected to GND, then the contents of the Sector Protection Register cannot be changed. If the $\overline{WP}$ pin is deasserted or permanently connected to V_{CC}, then the contents of the Sector Protection Register can be modified.

The $\overline{WP}$ pin will override the software controlled protection method but only for protecting the sectors.

Example: If the sectors were not previously protected by the Enable Sector Protection command, then simply asserting the $\overline{WP}$ pin would enable the sector protection within the maximum specified t_{WPE} time. When the $\overline{WP}$ pin is deasserted, however, the sector protection would no longer be enabled (after the maximum specified t_{WPD} time) as long as the Enable Sector Protection command was not issued while the $\overline{WP}$ pin was asserted. If the Enable Sector Protection command was issued before or while the $\overline{WP}$ pin was asserted, then simply deasserting the $\overline{WP}$ pin would not disable the sector protection. In this case, the Disable Sector Protection command would need to be issued while the $\overline{WP}$ pin is deasserted to disable the sector protection. The Disable Sector Protection command is also ignored whenever the $\overline{WP}$ pin is asserted.

A noise filter is incorporated to help protect against spurious noise that may inadvertently assert or deassert the $\overline{WP}$ pin.

<blue>Figures 7-3 and <blue>Table 7-3 detail the sector protection status for various scenarios of the $\overline{WP}$ pin, the Enable Sector Protection command, and the Disable Sector Protection command.

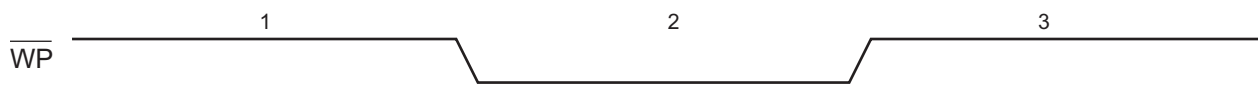**Figure 7-3.** $\overline{WP}$ **Pin and Protection Status**

Table 7-3. $\overline{WP}$ **Pin and Protection Status**

Time Period	$\overline{WP}$ Pin	Enable Sector Protection Command	Disable Sector Protection Command	Sector Protection Status	Sector Protection Register
1	High	Command Not Issued Previously	X	Disabled	Read/Write
		—	Issue Command	Disabled	Read/Write
		Issue Command	—	Enabled	Read/Write
2	Low	X	X	Enabled	Read
3	High	Command Issued During Period 1 or 2	Not Issued Yet	Enabled	Read/Write
		—	Issue Command	Disabled	Read/Write
		Issue Command	—	Enabled	Read/Write

7.3   Sector Protection Register

The nonvolatile Sector Protection Register specifies which sectors are to be protected or unprotected with either the software or hardware controlled protection methods. The Sector Protection Register contains 16 bytes of data, of which byte locations 0 through 15 contain values that specify whether Sectors 0 through 15 will be protected or unprotected. The Sector Protection Register is user modifiable and must be erased before it can be reprogrammed. <blue>Table 7-4 illustrates the format of the Sector Protection Register.

Table 7-4. Sector Protection Register

Sector Number	0 (0a, 0b)	1 to 15
Protected	See <blue>Table 7-5	FFh
Unprotected		00h

Note: 1.   The default values for bytes 0 through 15 are 00h when shipped from Adesto.

Table 7-5. Sector 0 (0a, 0b) Sector Protection Register Byte Value

	Bit 7:6	Bit 5:4	Bit 3:2	Bit 1:0	
	Sector 0a (Page 0-7)	Sector 0b (Page 8-255)	N/A	N/A	Data Value
Sectors 0a and 0b Unprotected	00	00	XX	XX	0xh
Protect Sector 0a	11	00	XX	XX	Cxh
Protect Sector 0b	00	11	XX	XX	3xh
Protect Sectors 0a and 0b	11	11	XX	XX	Fxh

Note: 1.   x = Don't care

7.3.1 Erase Sector Protection Register

In order to modify and change the values of the Sector Protection Register, it must first be erased using the Erase Sector Protection Register command.
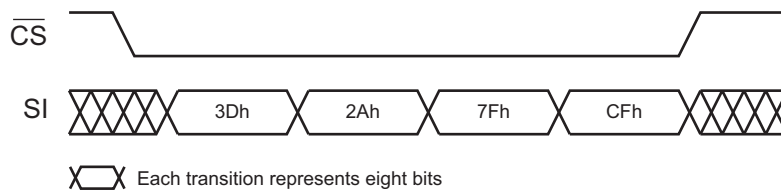
To erase the Sector Protection Register, a 4-byte command sequence of 3Dh, 2Ah, 7Fh, and CFh must be clocked into the device. After the last bit of the opcode sequence has been clocked in, the $\overline{CS}$ pin must be deasserted to initiate the internally self-timed erase cycle. The erasing of the Sector Protection Register should take place in a maximum time of t_{PE}. During this time, the RDY/$\overline{BUSY}$ bit in the Status Register will indicate that the device is busy. If the device is powered-down before the completion of the erase cycle, then the contents of the Sector Protection Register cannot be guaranteed.

The Sector Protection Register can be erased with sector protection enabled or disabled. Since the erased state (FFh) of each byte in the Sector Protection Register is used to indicate that a sector is specified for protection, leaving the sector protection enabled during the erasing of the register allows the protection scheme to be more effective in the prevention of accidental programming or erasing of the device. If for some reason an erroneous program or erase command is sent to the device immediately after erasing the Sector Protection Register and before the register can be reprogrammed, then the erroneous program or erase command will not be processed because all sectors would be protected.

Table 7-6.   Erase Sector Protection Register Command

Command	Byte 1	Byte 2	Byte 3	Byte 4
Erase Sector Protection Register	3Dh	2Ah	7Fh	CFh

Figure 7-4.   Erase Sector Protection Register

Each transition represents eight bits

7.3.2 Program Sector Protection Register

Once the Sector Protection Register has been erased, it can be reprogrammed using the Program Sector Protection Register command.

To program the Sector Protection Register, a 4-byte command sequence of 3Dh, 2Ah, 7Fh, and FCh must be clocked into the device followed by 16 bytes of data corresponding to Sectors 0 through 15. After the last bit of the opcode sequence and data have been clocked in, the $\overline{CS}$ pin must be deasserted to initiate the internally self-timed program cycle. The programming of the Sector Protection Register should take place in a maximum time of t_P. During this time, the RDY/$\overline{BUSY}$ bit in the Status Register will indicate that the device is busy. If the device is powered-down before the completion of the erase cycle, then the contents of the Sector Protection Register cannot be guaranteed.

If the proper number of data bytes is not clocked in before the $\overline{CS}$ pin is deasserted, then the protection status of the sectors corresponding to the bytes not clocked in cannot be guaranteed.

Example: If only the first two bytes are clocked in instead of the complete 16 bytes, then the protection status of the last 14 sectors cannot be guaranteed. Furthermore, if more than 16 bytes of data is clocked into the device, then the data will wrap back around to the beginning of the register. For instance, if 17 bytes of data are clocked in, then the 17th byte will be stored at byte location 0 of the Sector Protection Register.

The data bytes clocked into the Sector Protection Register need to be valid values (0xh, 3xh, Cxh, and Fxh for Sector 0a or Sector 0b, and 00h or FFh for other sectors) in order for the protection to function correctly. If a non-valid value is

clocked into a byte location of the Sector Protection Register, then the protection status of the sector corresponding to that byte location cannot be guaranteed.

Example: If a value of 17h is clocked into byte location 2 of the Sector Protection Register, then the protection status of Sector 2 cannot be guaranteed.
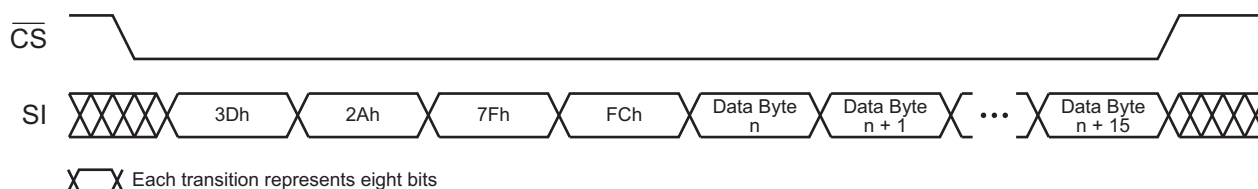
The Sector Protection Register can be reprogrammed while the sector protection is enabled or disabled. Being able to reprogram the Sector Protection Register with the sector protection enabled allows the user to temporarily disable the sector protection to an individual sector rather than disabling sector protection completely.

The Program Sector Protection Register command utilizes Buffer 1 for processing. Therefore, the contents of Buffer 1 will be altered from its previous state when this command is issued.

Table 7-7.   Program Sector Protection Register Command

Command	Byte 1	Byte 2	Byte 3	Byte 4
Program Sector Protection Register	3Dh	2Ah	7Fh	FCh

Figure 7-5.   Program Sector Protection Register

Each transition represents eight bits

7.3.3   Read Sector Protection Register

To read the Sector Protection Register, an opcode of 32h and three dummy bytes must be clocked into the device. After the last bit of the opcode and dummy bytes have been clocked in, any additional clock pulses on the SCK pin will result in the Sector Protection Register contents being output on the SO pin. The first byte (byte location 0) corresponds to Sector 0 (0a and 0b), the second byte corresponds to Sector 1, and the last byte (byte location 15) corresponds to Sector 15. Once the last byte of the Sector Protection Register has been clocked out, any additional clock pulses will result in undefined data being output on the SO pin. The CS pin must be deasserted to terminate the Read Sector Protection Register operation and put the output into a high-impedance state.

Table 7-8.   Read Sector Protection Register Command

Command	Byte 1	Byte 2	Byte 3	Byte 4
Read Sector Protection Register	32h	XXh	XXh	XXh

Note: 1.   XX = Dummy byte

Figure 7-6.   Read Sector Protection Register

Each transition represents eight bits