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

Features

- Single 2.7V - 3.6V Supply
- Serial Peripheral Interface (SPI) Compatible
  - Supports SPI Modes 0 and 3
  - Supports RapidS™ Operation
  - Supports Dual-Input Program and Dual-Output Read
- Very High Operating Frequencies
  - 100MHz for RapidS
  - 85MHz for SPI
  - Clock-to-Output (t_V) of 5ns Maximum
- Flexible, Optimized Erase Architecture for Code + Data Storage Applications
  - Uniform 4-Kbyte Block Erase
  - Uniform 32-Kbyte Block Erase
  - Uniform 64-Kbyte Block Erase
  - Full Chip Erase
- Individual Sector Protection with Global Protect/Unprotect Feature
  - 16 Sectors of 64-Kbytes Each
- Hardware Controlled Locking of Protected Sectors via $\overline{\text{WP}}$ Pin
- Sector Lockdown
  - Make Any Combination of 64-Kbyte Sectors Permanently Read-Only
- 128-Byte Programmable OTP Security Register
- Flexible Programming
  - Byte/Page Program (1- to 256-Bytes)
- Fast Program and Erase Times
  - 1.0ms Typical Page Program (256 Bytes) Time
  - 50ms Typical 4-Kbyte Block Erase Time
  - 250ms Typical 32-Kbyte Block Erase Time
  - 400ms Typical 64-Kbyte Block Erase Time
- Automatic Checking and Reporting of Erase/Program Failures
- Software Controlled Reset
- JEDEC Standard Manufacturer and Device ID Read Methodology
- Low Power Dissipation
  - 5mA Active Read Current (Typical at 20MHz)
  - 5µA Deep Power-Down Current (Typical)
- Endurance: 100,000 Program/Erase Cycles
- Data Retention: 20 Years
- Complies with Full Industrial Temperature Range
- Industry Standard Green (Pb/Halide-free/RoHS Compliant) Package Options
  - 8-lead SOIC (150-mil and 208-mil wide)
  - 8-pad Ultra Thin DFN (5 x 6 x 0.6mm)

8-Mbit
2.7V Minimum
SPI Serial Flash
Memory

AT25DF081A

8715C–SFLSH–1/2013

1. Description

The Adesto® AT25DF081A is a serial interface Flash memory device designed for use in a wide variety of high-volume consumer based applications in which program code is shadowed from Flash memory into embedded or external RAM for execution. The flexible erase architecture of the AT25DF081A, with its erase granularity as small as 4-Kbytes, makes it ideal for data storage as well, eliminating the need for additional data storage EEPROM devices.

The physical sectoring and the erase block sizes of the AT25DF081A have been optimized to meet the needs of today's code and data storage applications. By optimizing the size of the physical sectors and erase blocks, the memory space can be used much more efficiently. Because certain code modules and data storage segments must reside by themselves in their own protected sectors, the wasted and unused memory space that occurs with large sectored and large block erase Flash memory devices can be greatly reduced. This increased memory space efficiency allows additional code routines and data storage segments to be added while still maintaining the same overall device density.

The AT25DF081A also offers a sophisticated method for protecting individual sectors against erroneous or malicious program and erase operations. By providing the ability to individually protect and unprotect sectors, a system can unprotect a specific sector to modify its contents while keeping the remaining sectors of the memory array securely protected. This is useful in applications where program code is patched or updated on a subroutine or module basis, or in applications where data storage segments need to be modified without running the risk of errant modifications to the program code segments. In addition to individual sector protection capabilities, the AT25DF081A incorporates Global Protect and Global Unprotect features that allow the entire memory array to be either protected or unprotected all at once. This reduces overhead during the manufacturing process since sectors do not have to be unprotected one-by-one prior to initial programming.

To take code and data protection to the next level, the AT25DF081A incorporates a sector lockdown mechanism that allows any combination of individual 64-Kbyte sectors to be locked down and become permanently read-only. This addresses the need of certain secure applications that require portions of the Flash memory array to be permanently protected against malicious attempts at altering program code, data modules, security information, or encryption/decryption algorithms, keys, and routines. The device also contains a specialized OTP (One-Time Programmable) Security Register that can be used for purposes such as unique device serialization, system-level Electronic Serial Number (ESN) storage, locked key storage, etc.

Specifically designed for use in 3-volt systems, the AT25DF081A supports read, program, and erase operations with a supply voltage range of 2.7V to 3.6V. No separate voltage is required for programming and erasing.

2. Pin Descriptions and Pinouts

Table 2-1. Pin Descriptions

Symbol	Name and Function	Asserted State	Type
$\overline{\text{CS}}$	**CHIP SELECT:** Asserting the $\overline{\text{CS}}$ pin selects the device. When the $\overline{\text{CS}}$ pin is deasserted, the device will be deselected and normally be placed in standby mode (not Deep Power-Down mode), and the SO pin will be in a high-impedance state. When the device is deselected, data will not be accepted on the SI pin. A high-to-low transition on the $\overline{\text{CS}}$ 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 in 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 (SIO)	**SERIAL INPUT (SERIAL INPUT/OUTPUT):** 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 in on the rising edge of SCK. With the Dual-Output Read Array command, the SI pin becomes an output pin (SIO) to allow two bits of data (on the SO and SIO pins) to be clocked out on every falling edge of SCK. To maintain consistency with SPI nomenclature, the SIO pin will be referenced as SI throughout the document with exception to sections dealing with the Dual-Output Read Array command in which it will be referenced as SIO. Data present on the SI pin will be ignored whenever the device is deselected ($\overline{\text{CS}}$ is deasserted).	-	Input/Output
SO (SOI)	**SERIAL OUTPUT (SERIAL OUTPUT/INPUT):** 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-Input Byte/Page Program command, the SO pin becomes an input pin (SOI) to allow two bits of data (on the SOI and SI pins) to be clocked in on every rising edge of SCK. To maintain consistency with SPI nomenclature, the SOI pin will be referenced as SO throughout the document with exception to sections dealing with the Dual-Input Byte/Page Program command in which it will be referenced as SOI. The SO pin will be in a high-impedance state whenever the device is deselected ($\overline{\text{CS}}$ is deasserted).	-	Output/Input
$\overline{\text{WP}}$	**WRITE PROTECT:** The $\overline{\text{WP}}$ pin controls the hardware locking feature of the device. Please refer to "Protection Commands and Features" on page 17 for more details on protection features and the $\overline{\text{WP}}$ pin. The $\overline{\text{WP}}$ pin is internally pulled-high and may be left floating if hardware controlled protection will not be used. However, it is recommended that the $\overline{\text{WP}}$ pin also be externally connected to V_{CC} whenever possible.	Low	Input
$\overline{\text{HOLD}}$	**HOLD:** The $\overline{\text{HOLD}}$ pin is used to temporarily pause serial communication without deselecting or resetting the device. While the $\overline{\text{HOLD}}$ pin is asserted, transitions on the SCK pin and data on the SI pin will be ignored, and the SO pin will be in a high-impedance state. The $\overline{\text{CS}}$ pin must be asserted, and the SCK pin must be in the low state in order for a Hold condition to start. A Hold condition pauses serial communication only and does not have an effect on internally self-timed operations such as a program or erase cycle. Please refer to "Hold" on page 41 for additional details on the Hold operation. The $\overline{\text{HOLD}}$ pin is internally pulled-high and may be left floating if the Hold function will not be used. However, it is recommended that the $\overline{\text{HOLD}}$ pin also be externally connected to V_{CC} whenever possible.	Low	Input

Table 2-1. Pin Descriptions (Continued)

Symbol	Name and Function	Asserted State	Type
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.	-	Power

Figure 2-1. 8-SOIC (Top View)

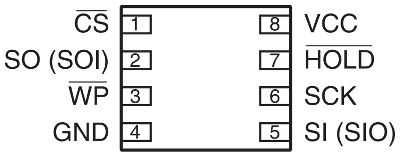
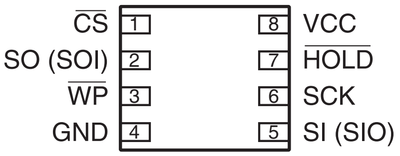
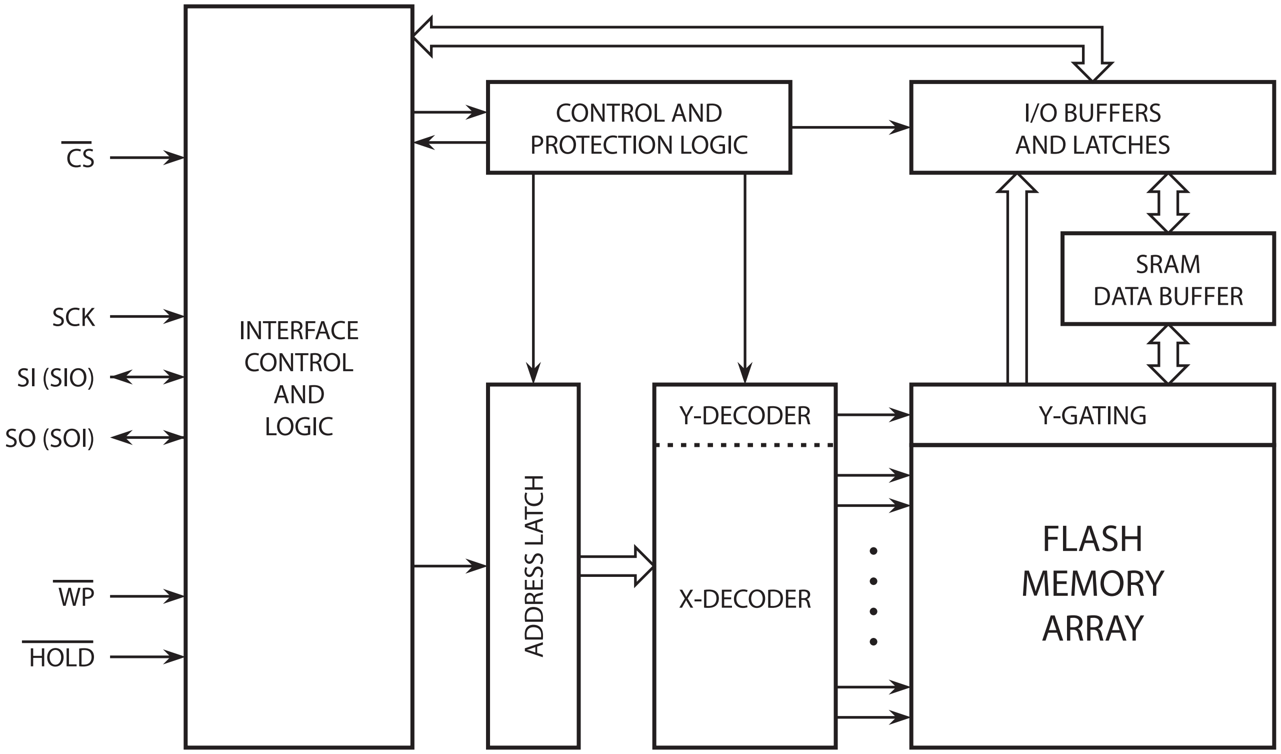


Figure 2-2. 8-UDFN (Top View)



3. Block Diagram

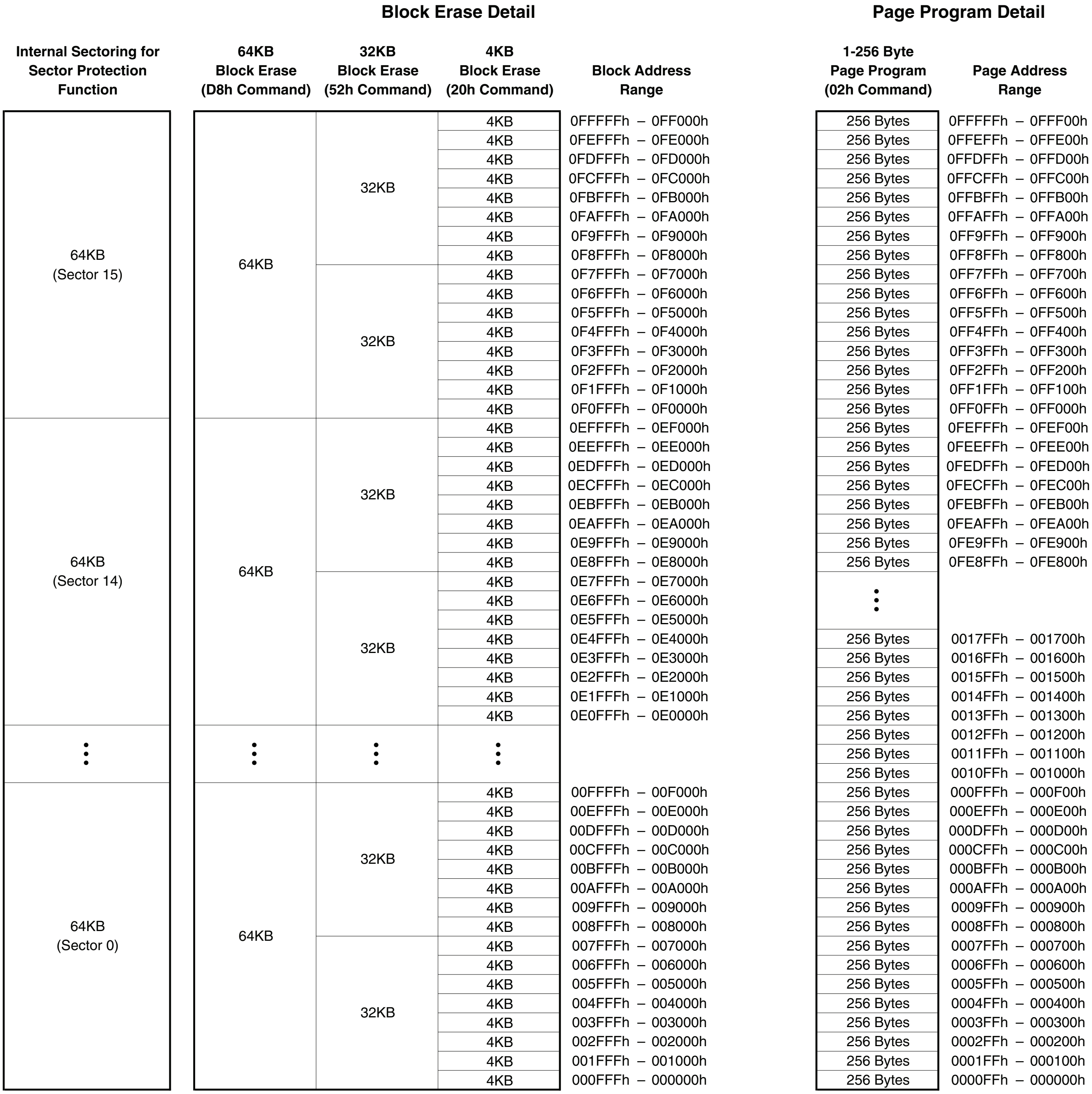
Figure 3-1. Block Diagram



4. Memory Array

To provide the greatest flexibility, the memory array of the AT25DF081A can be erased in four levels of granularity including a full chip erase. In addition, the array has been divided into physical sectors of uniform size, of which each sector can be individually protected from program and erase operations. The size of the physical sectors is optimized for both code and data storage applications, allowing both code and data segments to reside in their own isolated regions. The Memory Architecture Diagram illustrates the breakdown of each erase level as well as the breakdown of each physical sector.

Figure 4-1. Memory Architecture Diagram



Block Erase Detail

Internal Sectoring for Sector Protection Function	64KB Block Erase (D8h Command)	32KB Block Erase (52h Command)	4KB Block Erase (20h Command)	Block Address Range
64KB (Sector 15)	64KB	32KB	4KB	0FFFFFh – 0FF000h
			4KB	0FEFFFh – 0FE000h
			4KB	0FDFFFh – 0FD000h
			4KB	0FCFFFh – 0FC000h
			4KB	0FBFFFh – 0FB000h
			4KB	0FAFFFh – 0FA000h
			4KB	0F9FFFh – 0F9000h
			4KB	0F8FFFh – 0F8000h
		32KB	4KB	0F7FFFh – 0F7000h
			4KB	0F6FFFh – 0F6000h
			4KB	0F5FFFh – 0F5000h
			4KB	0F4FFFh – 0F4000h
			4KB	0F3FFFh – 0F3000h
			4KB	0F2FFFh – 0F2000h
			4KB	0F1FFFh – 0F1000h
			4KB	0F0FFFh – 0F0000h
64KB (Sector 14)	64KB	32KB	4KB	0EFFFFh – 0EF000h
			4KB	0EEFFFh – 0EE000h
			4KB	0EDFFFh – 0ED000h
			4KB	0ECFFFh – 0EC000h
			4KB	0EBFFFh – 0EB000h
			4KB	0EAFFFh – 0EA000h
			4KB	0E9FFFh – 0E9000h
			4KB	0E8FFFh – 0E8000h
		32KB	4KB	0E7FFFh – 0E7000h
			4KB	0E6FFFh – 0E6000h
			4KB	0E5FFFh – 0E5000h
			4KB	0E4FFFh – 0E4000h
			4KB	0E3FFFh – 0E3000h
			4KB	0E2FFFh – 0E2000h
			4KB	0E1FFFh – 0E1000h
			4KB	0E0FFFh – 0E0000h
⋮	⋮	⋮	⋮	
64KB (Sector 0)	64KB	32KB	4KB	00FFFFh – 00F000h
			4KB	00EFFFh – 00E000h
			4KB	00DFFFh – 00D000h
			4KB	00CFFFh – 00C000h
			4KB	00BFFFh – 00B000h
			4KB	00AFFFh – 00A000h
			4KB	009FFFh – 009000h
			4KB	008FFFh – 008000h
		32KB	4KB	007FFFh – 007000h
			4KB	006FFFh – 006000h
			4KB	005FFFh – 005000h
			4KB	004FFFh – 004000h
			4KB	003FFFh – 003000h
			4KB	002FFFh – 002000h
			4KB	001FFFh – 001000h
			4KB	000FFFh – 000000h

Page Program Detail

1-256 Byte Page Program (02h Command)	Page Address Range
256 Bytes	0FFFFFh – 0FFF00h
256 Bytes	0FFEFFh – 0FFE00h
256 Bytes	0FFDFFh – 0FFD00h
256 Bytes	0FFCFFh – 0FFC00h
256 Bytes	0FFBFFh – 0FFB00h
256 Bytes	0FFAFFh – 0FFA00h
256 Bytes	0FF9FFh – 0FF900h
256 Bytes	0FF8FFh – 0FF800h
256 Bytes	0FF7FFh – 0FF700h
256 Bytes	0FF6FFh – 0FF600h
256 Bytes	0FF5FFh – 0FF500h
256 Bytes	0FF4FFh – 0FF400h
256 Bytes	0FF3FFh – 0FF300h
256 Bytes	0FF2FFh – 0FF200h
256 Bytes	0FF1FFh – 0FF100h
256 Bytes	0FF0FFh – 0FF000h
256 Bytes	0FEFFFh – 0FEF00h
256 Bytes	0FEEFFh – 0FEE00h
256 Bytes	0FEDFFh – 0FED00h
256 Bytes	0FECFFh – 0FEC00h
256 Bytes	0FEBFFh – 0FEB00h
256 Bytes	0FEAFFh – 0FEA00h
256 Bytes	0FE9FFh – 0FE900h
256 Bytes	0FE8FFh – 0FE800h
⋮	
256 Bytes	0017FFh – 001700h
256 Bytes	0016FFh – 001600h
256 Bytes	0015FFh – 001500h
256 Bytes	0014FFh – 001400h
256 Bytes	0013FFh – 001300h
256 Bytes	0012FFh – 001200h
256 Bytes	0011FFh – 001100h
256 Bytes	0010FFh – 001000h
256 Bytes	000FFFh – 000F00h
256 Bytes	000EFFh – 000E00h
256 Bytes	000DFFh – 000D00h
256 Bytes	000CFFh – 000C00h
256 Bytes	000BFFh – 000B00h
256 Bytes	000AFFh – 000A00h
256 Bytes	0009FFh – 000900h
256 Bytes	0008FFh – 000800h
256 Bytes	0007FFh – 000700h
256 Bytes	0006FFh – 000600h
256 Bytes	0005FFh – 000500h
256 Bytes	0004FFh – 000400h
256 Bytes	0003FFh – 000300h
256 Bytes	0002FFh – 000200h
256 Bytes	0001FFh – 000100h
256 Bytes	0000FFh – 000000h

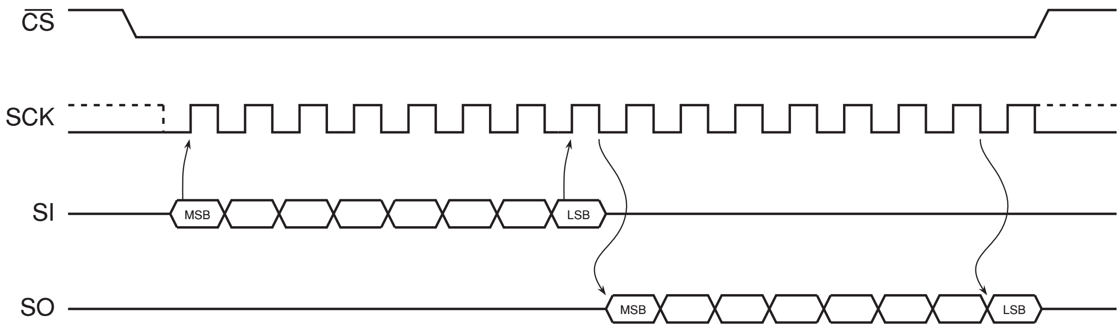
5. Device Operation

The AT25DF081A is controlled by a set of instructions that are sent from a host controller, commonly referred to as the SPI Master. The SPI Master communicates with the AT25DF081A via the SPI bus which is comprised of four signal lines: Chip Select ($\overline{\text{CS}}$), Serial Clock (SCK), Serial Input (SI), and Serial Output (SO).

The AT25DF081A features a dual-input program mode in which the SO pin becomes an input. Similarly, the device also features a dual-output read mode in which the SI pin becomes an output. In the Dual-Input Byte/Page Program command description, the SO pin will be referred to as the SOI (Serial Output/Input) pin, and in the Dual-Output Read Array command, the SI pin will be referenced as the SIO (Serial Input/Output) pin.

The SPI protocol defines a total of four modes of operation (mode 0, 1, 2, or 3) with each mode differing in respect to the SCK polarity and phase and how the polarity and phase control the flow of data on the SPI bus. The AT25DF081A supports the two most common modes, SPI Modes 0 and 3. The only difference between SPI Modes 0 and 3 is the polarity of the SCK signal when in the inactive state (when the SPI Master is in standby mode and not transferring any data). With SPI Modes 0 and 3, data is always latched in on the rising edge of SCK and always output on the falling edge of SCK.

Figure 5-1. SPI Mode 0 and 3



6. Commands and Addressing

A valid instruction or operation must always be started by first asserting the $\overline{\text{CS}}$ pin. After the $\overline{\text{CS}}$ pin has been asserted, the host controller must then clock out a valid 8-bit opcode on the SPI bus. Following the opcode, instruction dependent information such as address and data bytes would then be clocked out by the host controller. All opcode, address, and data bytes are transferred with the most-significant bit (MSB) first. An operation is ended by deasserting the $\overline{\text{CS}}$ pin.

Opcodes not supported by the AT25DF081A will be ignored by the device and no operation will be started. The device will continue to ignore any data presented on the SI pin until the start of the next operation ($\overline{\text{CS}}$ pin being deasserted and then reasserted). In addition, if the $\overline{\text{CS}}$ pin is deasserted before complete opcode and address information is sent to the device, then no operation will be performed and the device will simply return to the idle state and wait for the next operation.

Addressing of the device requires a total of three bytes of information to be sent, representing address bits A23-A0. Since the upper address limit of the AT25DF081A memory array is 0FFFFFh, address bits A23-A20 are always ignored by the device.

Table 6-1. Command Listing

Command	Opcode		Clock Frequency	Address Bytes	Dummy Bytes	Data Bytes
Read Commands						
Read Array	1Bh	0001 1011	Up to 100MHz	3	2	1+
	0Bh	0000 1011	Up to 85MHz	3	1	1+
	03h	0000 0011	Up to 50MHz	3	0	1+
Dual-Output Read Array	3Bh	0011 1011	Up to 85MHz	3	1	1+
Program and Erase Commands						
Block Erase (4 KBytes)	20h	0010 0000	Up to 100MHz	3	0	0
Block Erase (32 KBytes)	52h	0101 0010	Up to 100MHz	3	0	0
Block Erase (64 KBytes)	D8h	1101 1000	Up to 100MHz	3	0	0
Chip Erase	60h	0110 0000	Up to 100MHz	0	0	0
	C7h	1100 0111	Up to 100MHz	0	0	0
Byte/Page Program (1 to 256 Bytes)	02h	0000 0010	Up to 100MHz	3	0	1+
Dual-Input Byte/Page Program (1 to 256 Bytes)	A2h	1010 0010	Up to 100MHz	3	0	1+
Protection Commands						
Write Enable	06h	0000 0110	Up to 100MHz	0	0	0
Write Disable	04h	0000 0100	Up to 100MHz	0	0	0
Protect Sector	36h	0011 0110	Up to 100MHz	3	0	0
Unprotect Sector	39h	0011 1001	Up to 100MHz	3	0	0
Global Protect/Unprotect	Use Write Status Register Byte 1 Command					
Read Sector Protection Registers	3Ch	0011 1100	Up to 100MHz	3	0	1+
Security Commands						
Sector Lockdown	33h	0011 0011	Up to 100MHz	3	0	1
Freeze Sector Lockdown State	34h	0011 0100	Up to 100MHz	3	0	1
Read Sector Lockdown Registers	35h	0011 0101	Up to 100MHz	3	0	1+
Program OTP Security Register	9Bh	1001 1011	Up to 100MHz	3	0	1+
Read OTP Security Register	77h	0111 0111	Up to 100MHz	3	2	1+
Status Register Commands						
Read Status Register	05h	0000 0101	Up to 100MHz	0	0	1+
Write Status Register Byte 1	01h	0000 0001	Up to 100MHz	0	0	1
Write Status Register Byte 2	31h	0011 0001	Up to 100MHz	0	0	1
Miscellaneous Commands						
Reset	F0h	1111 0000	Up to 100MHz	0	0	1
Read Manufacturer and Device ID	9Fh	1001 1111	Up to 85MHz	0	0	1 to 4
Deep Power-Down	B9h	1011 1001	Up to 100MHz	0	0	0
Resume from Deep Power-Down	ABh	1010 1011	Up to 100MHz	0	0	0

7. Read Commands

7.1 Read Array

The Read Array command 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. The device incorporates an internal address counter that automatically increments on every clock cycle.

Three opcodes (1Bh, 0Bh, and 03h) can be used for the Read Array command. The use of each opcode depends on the maximum clock frequency that will be used to read data from the device. The 0Bh opcode can be used at any clock frequency up to the maximum specified by f_{CLK}, and the 03h opcode can be used for lower frequency read operations up to the maximum specified by f_{RDLF}. The 1Bh opcode allows the highest read performance possible and can be used at any clock frequency up to the maximum specified by f_{MAX}; however, use of the 1Bh opcode at clock frequencies above f_{CLK} should be reserved to systems employing the RapidS protocol.

To perform the Read Array operation, the $\overline{CS}$ pin must first be asserted and the appropriate opcode (1Bh, 0Bh, or 03h) must be clocked into the device. After the opcode has been clocked in, the three address bytes must be clocked in to specify the starting address location of the first byte to read within the memory array. Following the three address bytes, additional dummy bytes may need to be clocked into the device depending on which opcode is used for the Read Array operation. If the 1Bh opcode is used, then two dummy bytes must be clocked into the device after the three address bytes. If the 0Bh opcode is used, then a single dummy byte must be clocked in after the address bytes.

After the three address bytes (and the dummy bytes or byte if using opcodes 1Bh or 0Bh) have been clocked in, additional clock cycles will result in data being output on the SO pin. The data is always output with the MSB of a byte first. When the last byte (0FFFFFh) of the memory array has been read, the device will continue reading back at the beginning of the array (000000h). No delays will be incurred when wrapping around from the end of the array to the beginning of the array.

Deasserting the $\overline{CS}$ pin will terminate the read operation and put the SO pin into a high-impedance state. The $\overline{CS}$ pin can be deasserted at any time and does not require that a full byte of data be read.

Figure 7-1. Read Array – 1Bh Opcode

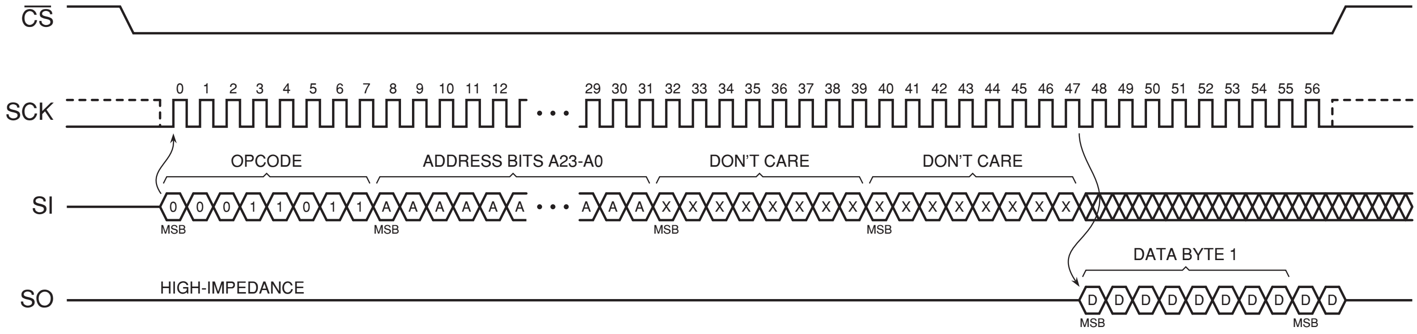


Figure 7-2. Read Array – 0Bh Opcode

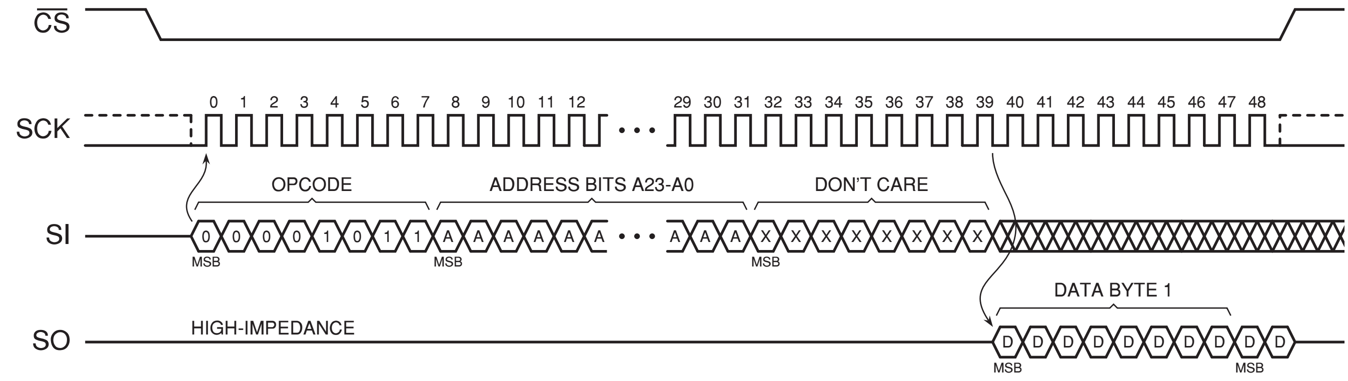
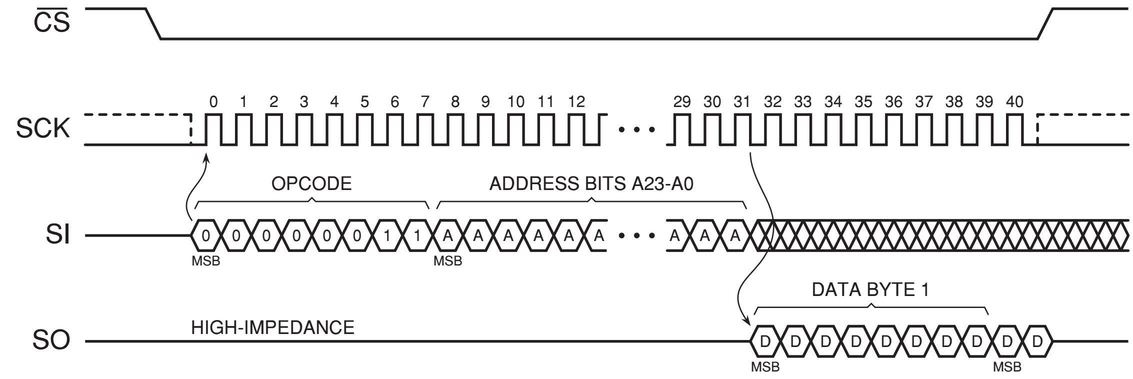


Figure 7-3. Read Array – 03h Opcode



7.2 Dual-Output Read Array

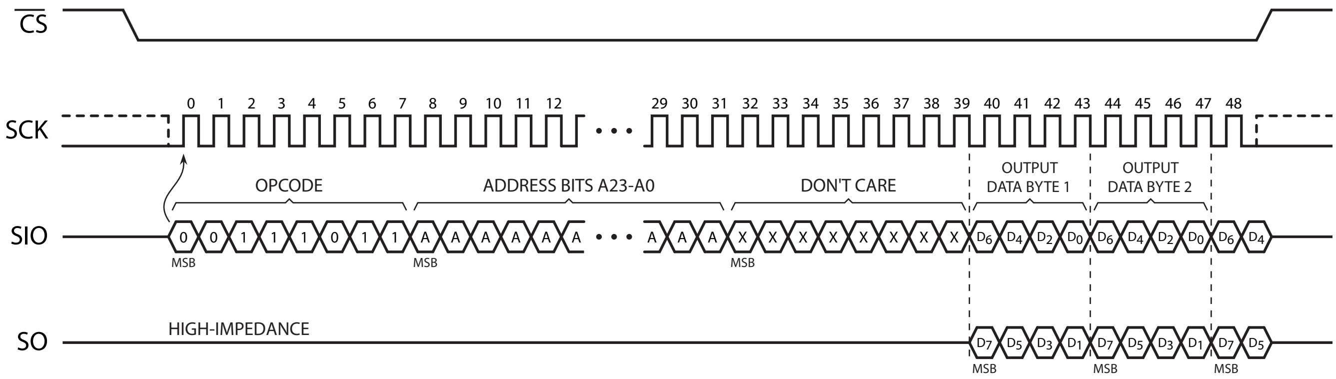
The Dual-Output Read Array command is similar to the standard Read Array command and can be used to sequentially read a continuous stream of data from the device by simply providing the clock signal once the initial starting address has been specified. Unlike the standard Read Array command, however, 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_{RDDO}. To perform the Dual-Output Read Array operation, the $\overline{CS}$ pin must first be asserted and the opcode of 3Bh must be clocked into the device. After the opcode has been clocked in, the three address bytes must be clocked in to specify the starting address location of the first byte to read within the memory array. Following the three address bytes, a single dummy byte must also be clocked into the device.

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 SO and SIO pins. The data is always output with the MSB of a byte first, and the MSB is always output on the SO pin. During the first clock cycle, bit seven of the first data byte will be output on the SO pin while bit six of the same data byte will be output on the SIO pin. During the next clock cycle, bits five and four of the first data byte will be output on the SO and SIO pins, respectively. The sequence continues with each byte of data being output after every four clock cycles. When the last byte (0FFFFFh) of the memory array has been read, the device will continue reading back at the beginning of the array (000000h). No delays will be incurred when wrapping around from the end of the array to the beginning of the array.

Deasserting the $\overline{CS}$ pin will terminate the read operation and put the SO and SIO pins into a high-impedance state. The $\overline{CS}$ pin can be deasserted at any time and does not require that a full byte of data be read.

Figure 7-4. Dual-Output Read Array



8. Program and Erase Commands

8.1 Byte/Page Program

The Byte/Page Program command allows anywhere from a single byte of data to 256-bytes of data to be programmed into previously erased memory locations. An erased memory location is one that has all eight bits set to the logical “1” state (a byte value of FFh). Before a Byte/Page Program command can be started, the Write Enable command must have been previously issued to the device (see “Write Enable” on page 17) to set the Write Enable Latch (WEL) bit of the Status Register to a logical “1” state.

To perform a Byte/Page Program command, an opcode of 02h must be clocked into the device followed by the three address bytes denoting the first byte location of the memory array to begin programming at. After the address bytes have been clocked in, data can then be clocked into the device and will be stored in an internal buffer.

If the starting memory address denoted by A23-A0 does not fall on an even 256-byte page boundary (A7-A0 are not all 0), then special circumstances regarding which memory locations to be programmed will apply. In this situation, any data that is sent to the device that goes beyond the end of the page will wrap around back to the beginning of the same page. For example, if the starting address denoted by A23-A0 is 0000FEh, and three bytes of data are sent to the device, then the first two bytes of data will be programmed at addresses 0000FEh and 0000FFh while the last byte of data will be programmed at address 000000h. The remaining bytes in the page (addresses 000001h through 0000FDh) will not be programmed and will remain in the erased state (FFh). In addition, if more than 256-bytes of data are sent to the device, then only the last 256-bytes sent will be latched into the internal buffer.

When the $\overline{CS}$ pin is deasserted, the device will take the data stored in the internal buffer and program it into the appropriate memory array locations based on the starting address specified by A23-A0 and the number of data bytes sent to the device. If less than 256-bytes of data were sent to the device, then the remaining bytes within the page will not be programmed and will remain in the erased state (FFh). The programming of the data bytes is internally self-timed and should take place in a time of t_{PP} or t_{BP} if only programming a single byte.

The three address bytes and at least one complete byte of data must be clocked into the device before the $\overline{CS}$ pin is deasserted, and the $\overline{CS}$ pin must be deasserted on even byte boundaries (multiples of eight bits); otherwise, the device will abort the operation and no data will be programmed into the memory array. In addition, if the address specified by A23-A0 points to a memory location within a sector that is in the protected state (see “Protect Sector” on page 19) or locked down (see “Sector Lockdown” on page 25), then the Byte/Page Program command will not be executed, and the device will return to the idle state once the $\overline{CS}$ pin has been deasserted. The WEL bit in the Status Register will be reset back to the logical “0” state if the program cycle aborts due to an incomplete address being sent, an incomplete byte of data being sent, the $\overline{CS}$ pin being deasserted on uneven byte boundaries, or because the memory location to be programmed is protected or locked down.

While the device is programming, the Status Register can be read and will indicate that the device is busy. For faster throughput, it is recommended that the Status Register be polled rather than waiting the t_{BP} or t_{PP} time to determine if the data bytes have finished programming. At some point before the program cycle completes, the WEL bit in the Status Register will be reset back to the logical “0” state.

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.

Figure 8-1. Byte Program

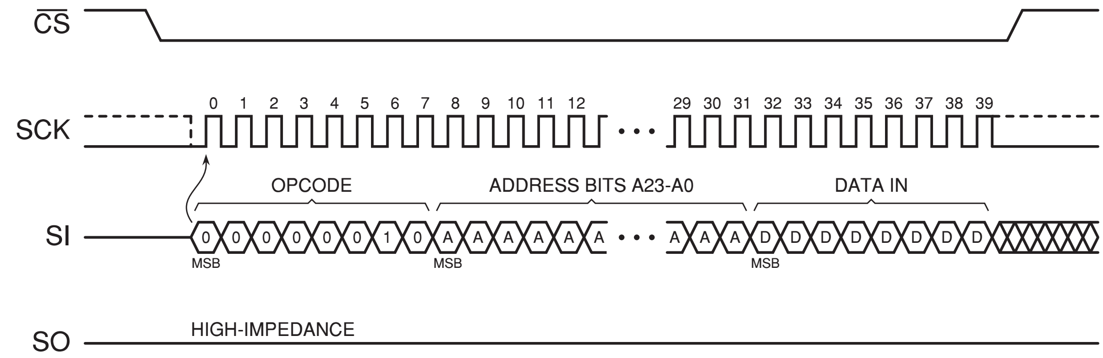
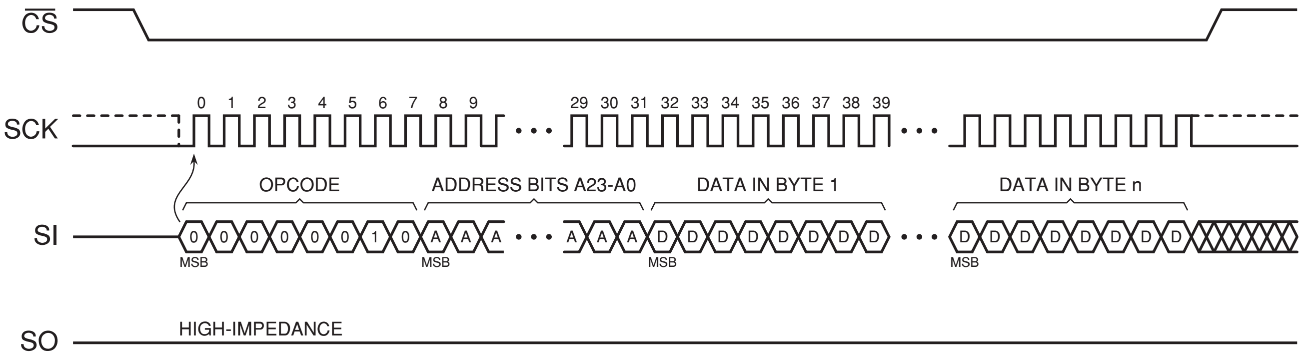


Figure 8-2. Page Program



8.2 Dual-Input Byte/Page Program

The Dual-Input Byte/Page Program command is similar to the standard Byte/Page Program command and can be used to program anywhere from a single byte of data up to 256-bytes of data into previously erased memory locations. Unlike the standard Byte/Page Program command, however, the Dual-Input Byte/Page Program command allows two bits of data to be clocked into the device on every clock cycle rather than just one.

Before the Dual-Input Byte/Page Program command can be started, the Write Enable command must have been previously issued to the device (see "Write Enable" on page 17) to set the Write Enable Latch (WEL) bit of the Status Register to a logical "1" state. To perform a Dual-Input Byte/Page Program command, an opcode of A2h must be clocked into the device followed by the three address bytes denoting the first byte location of the memory array to begin programming at. After the address bytes have been clocked in, data can then be clocked into the device two bits at a time on both the SOI and SI pins.

The data is always input with the MSB of a byte first, and the MSB is always input on the SOI pin. During the first clock cycle, bit seven of the first data byte would be input on the SOI pin while bit six of the same data byte would be input on the SI pin. During the next clock cycle, bits five and four of the first data byte would be input on the SOI and SI pins, respectively. The sequence would continue with each byte of data being input after every four clock cycles. Like the standard Byte/Page Program command, all data clocked into the device is stored in an internal buffer.

If the starting memory address denoted by A23-A0 does not fall on an even 256-byte page boundary (A7-A0 are not all 0), then special circumstances regarding which memory locations to be programmed will apply. In this situation, any data that is sent to the device that goes beyond the end of the page will wrap around back to the beginning of the same page. For example, if the starting address denoted by A23-A0 is 0000FEh, and three bytes of data are sent to the device, then the first two bytes of data will be programmed at addresses 0000FEh and 0000FFh while the last byte of data will be programmed at address 000000h. The remaining bytes in the page (addresses 000001h through 0000FDh) will not be programmed and will remain in the erased state (FFh). In addition, if more than 256-bytes of data are sent to the device, then only the last 256-bytes sent will be latched into the internal buffer.

When the $\overline{CS}$ pin is deasserted, the device will take the data stored in the internal buffer and program it into the appropriate memory array locations based on the starting address specified by A23-A0 and the number of data bytes sent to the device. If less than 256-bytes of data were sent to the device, then the remaining bytes within the page will not be programmed and will remain in the erased state (FFh). The programming of the data bytes is internally self-timed and should take place in a time of t_{PP} or t_{BP} if only programming a single byte.

The three address bytes and at least one complete byte of data must be clocked into the device before the $\overline{CS}$ pin is deasserted, and the $\overline{CS}$ pin must be deasserted on even byte boundaries (multiples of eight bits); otherwise, the device will abort the operation and no data will be programmed into the memory array. In addition, if the address specified by A23-A0 points to a memory location within a sector that is in the protected state (see "Protect Sector" on page 19) or locked down (see "Sector Lockdown" on page 25), then the Byte/Page Program command will not be executed, and the device will return to the idle state once the $\overline{CS}$ pin has been deasserted. The WEL bit in the Status Register will be reset back to the logical "0" state if the program cycle aborts due to an incomplete address being sent, an incomplete byte of data being sent, the $\overline{CS}$ pin being deasserted on uneven byte boundaries, or because the memory location to be programmed is protected or locked down.

While the device is programming, the Status Register can be read and will indicate that the device is busy. For faster throughput, it is recommended that the Status Register be polled rather than waiting the t_{BP} or t_{PP} time to determine if the data bytes have finished programming. At some point before the program cycle completes, the WEL bit in the Status Register will be reset back to the logical "0" state.

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.

Figure 8-3. Dual-Input Byte Program

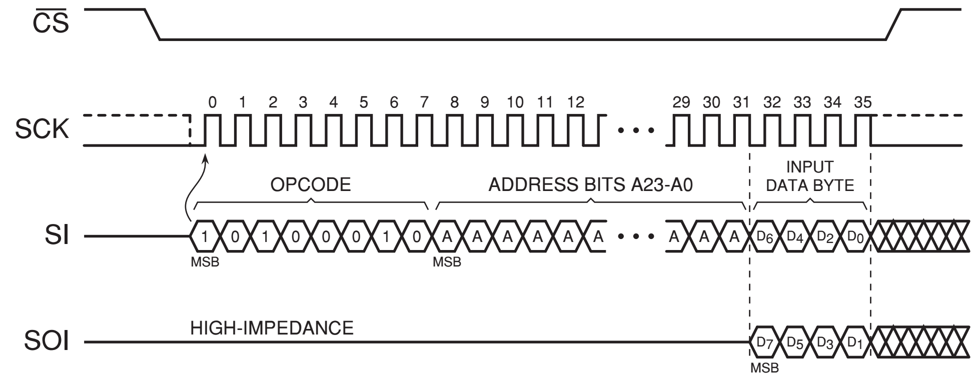
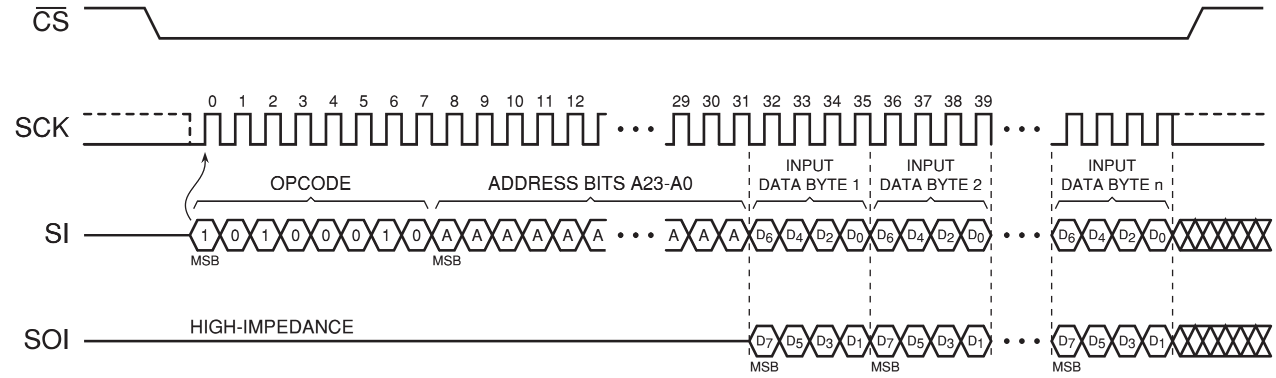


Figure 8-4. Dual-Input Page Program



8.3 Block Erase

A block of 4-, 32-, or 64-Kbytes can be erased (all bits set to the logical "1" state) in a single operation by using one of three different opcodes for the Block Erase command. An opcode of 20h is used for a 4-Kbyte erase, an opcode of 52h is used for a 32-Kbyte erase, and an opcode of D8h is used for a 64-Kbyte erase. Before a Block Erase command can be started, the Write Enable command must have been previously issued to the device to set the WEL bit of the Status Register to a logical "1" state.

To perform a Block Erase, the $\overline{\text{CS}}$ pin must first be asserted and the appropriate opcode (20h, 52h, or D8h) must be clocked into the device. After the opcode has been clocked in, the three address bytes specifying an address within the 4-, 32-, or 64-Kbyte block to be erased must be clocked in. Any additional data clocked into the device will be ignored. When the $\overline{\text{CS}}$ pin is deasserted, the device will erase the appropriate block. The erasing of the block is internally self-timed and should take place in a time of t_{BLKE}.

Since the Block Erase command erases a region of bytes, the lower order address bits do not need to be decoded by the device. Therefore, for a 4-Kbyte erase, address bits A11-A0 will be ignored by the device and their values can be either a logical "1" or "0". For a 32-Kbyte erase, address bits A14-A0 will be ignored, and for a 64-Kbyte erase, address bits A15-A0 will be ignored by the device. Despite the lower order address bits not being decoded by the device, the complete three address bytes must still be clocked into the device before the $\overline{\text{CS}}$ pin is deasserted, and the $\overline{\text{CS}}$ pin must be deasserted on an even byte boundary (multiples of eight bits); otherwise, the device will abort the operation and no erase operation will be performed.

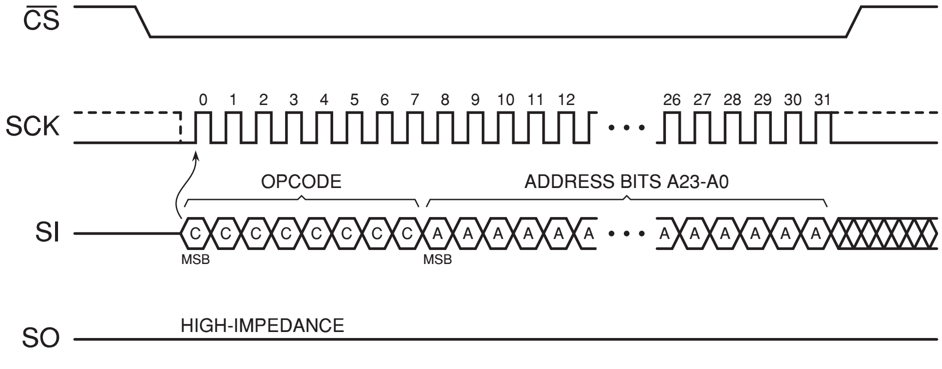
If the address specified by A23-A0 points to a memory location within a sector that is in the protected or locked down state, then the Block Erase command will not be executed, and the device will return to the idle state once the $\overline{\text{CS}}$ pin has been deasserted.

The WEL bit in the Status Register will be reset back to the logical "0" state if the erase cycle aborts due to an incomplete address being sent, the $\overline{\text{CS}}$ pin being deasserted on uneven byte boundaries, or because a memory location within the region to be erased is protected or locked down.

While the device is executing a successful erase cycle, the Status Register can be read and will indicate that the device is busy. For faster throughput, it is recommended that the Status Register be polled rather than waiting the t_{BLKE} time to determine if the device has finished erasing. At some point before the erase cycle completes, the WEL bit in the Status Register will be reset back to the logical "0" state.

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

Figure 8-5. Block Erase



8.4 Chip Erase

The entire memory array can be erased in a single operation by using the Chip Erase command. Before a Chip Erase command can be started, the Write Enable command must have been previously issued to the device to set the WEL bit of the Status Register to a logical “1” state.

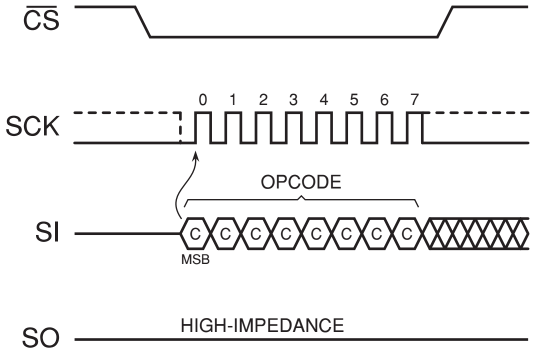
Two opcodes, 60h and C7h, can be used for the Chip Erase command. There is no difference in device functionality when utilizing the two opcodes, so they can be used interchangeably. To perform a Chip Erase, one of the two opcodes (60h or C7h) must be clocked into the device. Since the entire memory array 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. When the $\overline{\text{CS}}$ pin is deasserted, the device will erase the entire memory array. The erasing of the device is internally self-timed and should take place in a time of t_{CHPE}.

The complete opcode must be clocked into the device before the $\overline{\text{CS}}$ pin is deasserted, and the $\overline{\text{CS}}$ pin must be deasserted on an even byte boundary (multiples of eight bits); otherwise, no erase will be performed. In addition, if any sector of the memory array is in the protected or locked down state, then the Chip Erase command will not be executed, and the device will return to the idle state once the $\overline{\text{CS}}$ pin has been deasserted. The WEL bit in the Status Register will be reset back to the logical “0” state if the $\overline{\text{CS}}$ pin is deasserted on uneven byte boundaries or if a sector is in the protected or locked down state.

While the device is executing a successful erase cycle, the Status Register can be read and will indicate that the device is busy. For faster throughput, it is recommended that the Status Register be polled rather than waiting the t_{CHPE} time to determine if the device has finished erasing. At some point before the erase cycle completes, the WEL bit in the Status Register will be reset back to the logical “0” state.

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

Figure 8-6. Chip Erase



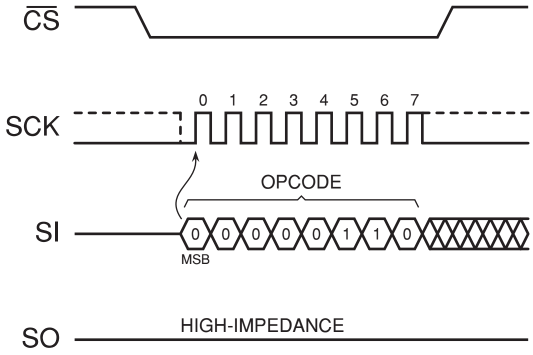
9. Protection Commands and Features

9.1 Write Enable

The Write Enable command is used to set the Write Enable Latch (WEL) bit in the Status Register to a logical “1” state. The WEL bit must be set before a Byte/Page Program, erase, Protect Sector, Unprotect Sector, Sector Lockdown, Freeze Sector Lockdown State, Program OTP Security Register, or Write Status Register command can be executed. This makes the issuance of these commands a two step process, thereby reducing the chances of a command being accidentally or erroneously executed. If the WEL bit in the Status Register is not set prior to the issuance of one of these commands, then the command will not be executed.

To issue the Write Enable command, the $\overline{\text{CS}}$ pin must first be asserted and the opcode of 06h 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 $\overline{\text{CS}}$ pin is deasserted, the WEL bit in the Status Register will be set to a logical “1”. The complete opcode must be clocked into the device before the $\overline{\text{CS}}$ pin is deasserted, and the $\overline{\text{CS}}$ pin must be deasserted on an even byte boundary (multiples of eight bits); otherwise, the device will abort the operation and the state of the WEL bit will not change.

Figure 9-1. Write Enable

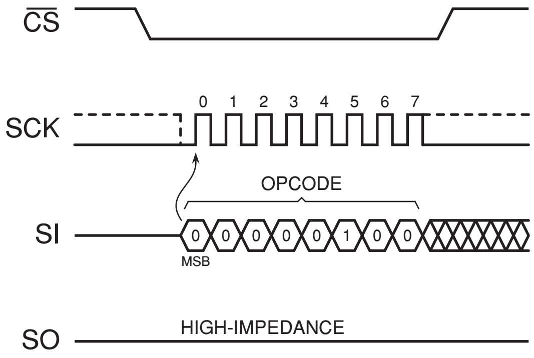


9.2 Write Disable

The Write Disable command is used to reset the Write Enable Latch (WEL) bit in the Status Register to the logical "0" state. With the WEL bit reset, all Byte/Page Program, erase, Protect Sector, Unprotect Sector, Sector Lockdown, Freeze Sector Lockdown State, Program OTP Security Register, and Write Status Register commands will not be executed. Other conditions can also cause the WEL bit to be reset; for more details, refer to the WEL bit section of the Status Register description.

To issue the Write Disable command, the $\overline{\text{CS}}$ pin must first be asserted and the opcode of 04h 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 $\overline{\text{CS}}$ pin is deasserted, the WEL bit in the Status Register will be reset to a logical “0”. The complete opcode must be clocked into the device before the $\overline{\text{CS}}$ pin is deasserted, and the $\overline{\text{CS}}$ pin must be deasserted on an even byte boundary (multiples of eight bits); otherwise, the device will abort the operation and the state of the WEL bit will not change.

Figure 9-2. Write Disable



9.3 Protect Sector

Every physical 64-Kbyte sector of the device has a corresponding single-bit Sector Protection Register that is used to control the software protection of a sector. Upon device power-up, each Sector Protection Register will default to the logical “1” state indicating that all sectors are protected and cannot be programmed or erased.

Issuing the Protect Sector command to a particular sector address will set the corresponding Sector Protection Register to the logical “1” state. The following table outlines the two states of the Sector Protection Registers.

Table 9-1. Sector Protection Register Values

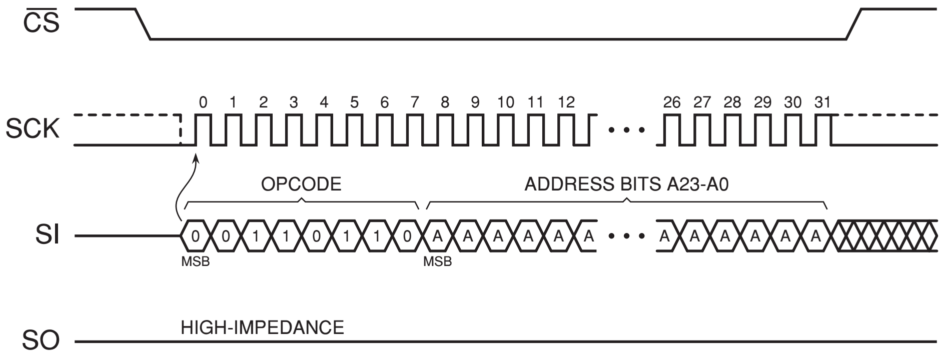
Value	Sector Protection Status
0	Sector is unprotected and can be programmed and erased.
1	Sector is protected and cannot be programmed or erased. This is the default state.

Before the Protect Sector command can be issued, the Write Enable command must have been previously issued to set the WEL bit in the Status Register to a logical “1”. To issue the Protect Sector command, the $\overline{CS}$ pin must first be asserted and the opcode of 36h must be clocked into the device followed by three address bytes designating any address within the sector to be protected. Any additional data clocked into the device will be ignored. When the $\overline{CS}$ pin is deasserted, the Sector Protection Register corresponding to the physical sector addressed by A23-A0 will be set to the logical “1” state, and the sector itself will then be protected from program and erase operations. In addition, the WEL bit in the Status Register will be reset back to the logical “0” state.

The complete three address bytes must be clocked into the device before the $\overline{CS}$ pin is deasserted, and the $\overline{CS}$ pin must be deasserted on an even byte boundary (multiples of eight bits); otherwise, the device will abort the operation. When the device aborts the Protect Sector operation, the state of the Sector Protection Register will be unchanged, and the WEL bit in the Status Register will be reset to a logical “0”.

As a safeguard against accidental or erroneous protecting or unprotecting of sectors, the Sector Protection Registers can themselves be locked from updates by using the SPRL (Sector Protection Registers Locked) bit of the Status Register (please refer to the Status Register description for more details). If the Sector Protection Registers are locked, then any attempts to issue the Protect Sector command will be ignored, and the device will reset the WEL bit in the Status Register back to a logical “0” and return to the idle state once the $\overline{CS}$ pin has been deasserted.

Figure 9-3. Protect Sector



9.4 Unprotect Sector

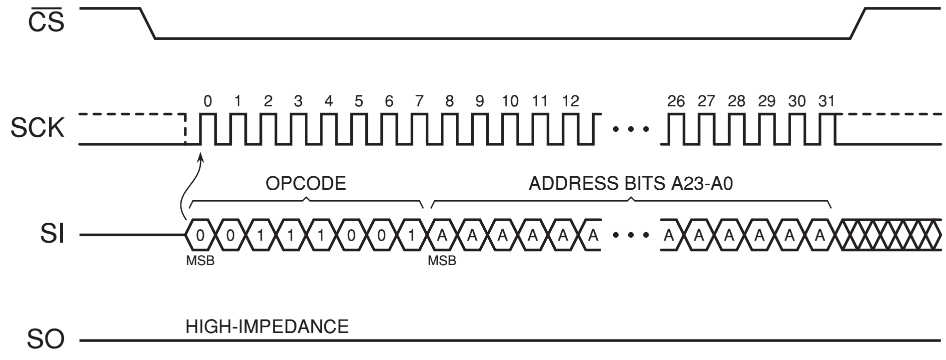
Issuing the Unprotect Sector command to a particular sector address will reset the corresponding Sector Protection Register to the logical "0" state (see Table 9-1 for Sector Protection Register values). Every physical sector of the device has a corresponding single-bit Sector Protection Register that is used to control the software protection of a sector.

Before the Unprotect Sector command can be issued, the Write Enable command must have been previously issued to set the WEL bit in the Status Register to a logical "1". To issue the Unprotect Sector command, the $\overline{\text{CS}}$ pin must first be asserted and the opcode of 39h must be clocked into the device. After the opcode has been clocked in, the three address bytes designating any address within the sector to be unprotected must be clocked in. Any additional data clocked into the device after the address bytes will be ignored. When the $\overline{\text{CS}}$ pin is deasserted, the Sector Protection Register corresponding to the sector addressed by A23-A0 will be reset to the logical "0" state, and the sector itself will be unprotected. In addition, the WEL bit in the Status Register will be reset back to the logical "0" state.

The complete three address bytes must be clocked into the device before the $\overline{\text{CS}}$ pin is deasserted, and the $\overline{\text{CS}}$ pin must be deasserted on an even byte boundary (multiples of eight bits); otherwise, the device will abort the operation, the state of the Sector Protection Register will be unchanged, and the WEL bit in the Status Register will be reset to a logical "0".

As a safeguard against accidental or erroneous locking or unlocking of sectors, the Sector Protection Registers can themselves be locked from updates by using the SPRL (Sector Protection Registers Locked) bit of the Status Register (please refer to the Status Register description for more details). If the Sector Protection Registers are locked, then any attempts to issue the Unprotect Sector command will be ignored, and the device will reset the WEL bit in the Status Register back to a logical "0" and return to the idle state once the $\overline{\text{CS}}$ pin has been deasserted.

Figure 9-4. Unprotect Sector



9.5 Global Protect/Unprotect

The Global Protect and Global Unprotect features can work in conjunction with the Protect Sector and Unprotect Sector functions. For example, a system can globally protect the entire memory array and then use the Unprotect Sector command to individually unprotect certain sectors and individually reprotect them later by using the Protect Sector command. Likewise, a system can globally unprotect the entire memory array and then individually protect certain sectors as needed.

Performing a Global Protect or Global Unprotect is accomplished by writing a certain combination of data to the Status Register using the Write Status Register Byte 1 command (see "Write Status Register Byte 1" on page 35 for command execution details). The Write Status Register command is also used to modify the SPRL (Sector Protection Registers Locked) bit to control hardware and software locking.

To perform a Global Protect, the appropriate $\overline{\text{WP}}$ pin and SPRL conditions must be met, and the system must write a logical "1" to bits five, four, three, and two of the first byte of the Status Register. Conversely, to perform a Global Unprotect, the same $\overline{\text{WP}}$ and SPRL conditions must be met but the system must write a logical "0" to bits five, four, three, and two of the first byte of the Status Register. Table 9-2 details the conditions necessary for a Global Protect or Global Unprotect to be performed.

Table 9-2. Valid SPRL and Global Protect/Unprotect Conditions

$\overline{\text{WP}}$ State	Current SPRL Value	New Write Status Register Byte 1 Data	Protection Operation	New SPRL Value
		Bit 7 6 5 4 3 2 1 0		
0	0	0 x 0 0 0 0 x x	Global Unprotect – all Sector Protection Registers reset to 0	0
		0 x 0 0 0 1 x x	No change to current protection.	0
			No change to current protection.	0
		0 x 1 1 1 0 x x	No change to current protection.	0
		0 x 1 1 1 1 x x	Global Protect – all Sector Protection Registers set to 1	0
		1 x 0 0 0 0 x x	Global Unprotect – all Sector Protection Registers reset to 0	1
		1 x 0 0 0 1 x x	No change to current protection.	1
			No change to current protection.	1
		1 x 1 1 1 0 x x	No change to current protection.	1
		1 x 1 1 1 1 x x	Global Protect – all Sector Protection Registers set to 1	1
0	1	x x x x x x x x	No change to the current protection level. All sectors currently protected will remain protected and all sectors currently unprotected will remain unprotected. The Sector Protection Registers are hard-locked and cannot be changed when the $\overline{\text{WP}}$ pin is LOW and the current state of SPRL is 1. Therefore, a Global Protect/Unprotect will not occur. In addition, the SPRL bit cannot be changed (the $\overline{\text{WP}}$ pin must be HIGH in order to change SPRL back to a 0).	
1	0	0 x 0 0 0 0 x x	Global Unprotect – all Sector Protection Registers reset to 0	0
		0 x 0 0 0 1 x x	No change to current protection.	0
			No change to current protection.	0
		0 x 1 1 1 0 x x	No change to current protection.	0
		0 x 1 1 1 1 x x	Global Protect – all Sector Protection Registers set to 1	0
		1 x 0 0 0 0 x x	Global Unprotect – all Sector Protection Registers reset to 0	1
		1 x 0 0 0 1 x x	No change to current protection.	1
			No change to current protection.	1
		1 x 1 1 1 0 x x	No change to current protection.	1
		1 x 1 1 1 1 x x	Global Protect – all Sector Protection Registers set to 1	1

Table 9-2. Valid SPRL and Global Protect/Unprotect Conditions (Continued)

$\overline{\text{WP}}$ State	Current SPRL Value	New Write Status Register Byte 1 Data Bit 7 6 5 4 3 2 1 0	Protection Operation	New SPRL Value
1	1	0 x 0 0 0 0 x x	No change to the current protection level. All sectors currently protected will remain protected, and all sectors currently unprotected will remain unprotected.	0
		0 x 0 0 0 1 x x		0
				0
		0 x 1 1 1 0 x x		0
		0 x 1 1 1 1 x x		0
		1 x 0 0 0 0 x x	The Sector Protection Registers are soft-locked and cannot be changed when the current state of SPRL is 1. Therefore, a Global Protect/Unprotect will not occur. However, the SPRL bit can be changed back to a 0 from a 1 since the $\overline{\text{WP}}$ pin is HIGH. To perform a Global Protect/Unprotect, the Write Status Register command must be issued again after the SPRL bit has been changed from a 1 to a 0.	1
		1 x 0 0 0 1 x x		1
				1
		1 x 1 1 1 0 x x		1
		1 x 1 1 1 1 x x		1

Essentially, if the SPRL bit of the Status Register is in the logical “0” state (Sector Protection Registers are not locked), then writing a 00h to the first byte of the Status Register will perform a Global Unprotect without changing the state of the SPRL bit. Similarly, writing a 7Fh to the first byte of the Status Register will perform a Global Protect and keep the SPRL bit in the logical “0” state. The SPRL bit can, of course, be changed to a logical “1” by writing an FFh if software-locking or hardware-locking is desired along with the Global Protect.

If the desire is to only change the SPRL bit without performing a Global Protect or Global Unprotect, then the system can simply write a 0Fh to the first byte of the Status Register to change the SPRL bit from a logical “1” to a logical “0” provided the $\overline{\text{WP}}$ pin is deasserted. Likewise, the system can write an F0h to change the SPRL bit from a logical “0” to a logical “1” without affecting the current sector protection status (no changes will be made to the Sector Protection Registers).

When writing to the first byte of the Status Register, bits five, four, three, and two will not actually be modified but will be decoded by the device for the purposes of the Global Protect and Global Unprotect functions. Only bit seven, the SPRL bit, will actually be modified. Therefore, when reading the first byte of the Status Register, bits five, four, three, and two will not reflect the values written to them but will instead indicate the status of the $\overline{\text{WP}}$ pin and the sector protection status. Please refer to “Read Status Register” on page 31 and Table 11-1 on page 31 for details on the Status Register format and what values can be read for bits five, four, three, and two.

9.6 Read Sector Protection Registers

The Sector Protection Registers can be read to determine the current software protection status of each sector. Reading the Sector Protection Registers, however, will not determine the status of the $\overline{\text{WP}}$ pin.

To read the Sector Protection Register for a particular sector, the $\overline{\text{CS}}$ pin must first be asserted and the opcode of 3Ch must be clocked in. Once the opcode has been clocked in, three address bytes designating any address within the sector must be clocked in. After the last address byte has been clocked in, the device will begin outputting data on the SO pin during every subsequent clock cycle. The data being output will be a repeating byte of either FFh or 00h to denote the value of the appropriate Sector Protection Register.

At clock frequencies above f_{CLK}, the first byte of data output will not be valid. Therefore, if operating at clock frequencies above f_{CLK}, at least two bytes of data must be clocked out from the device in order to determine the correct status of the appropriate Sector Protection Register.

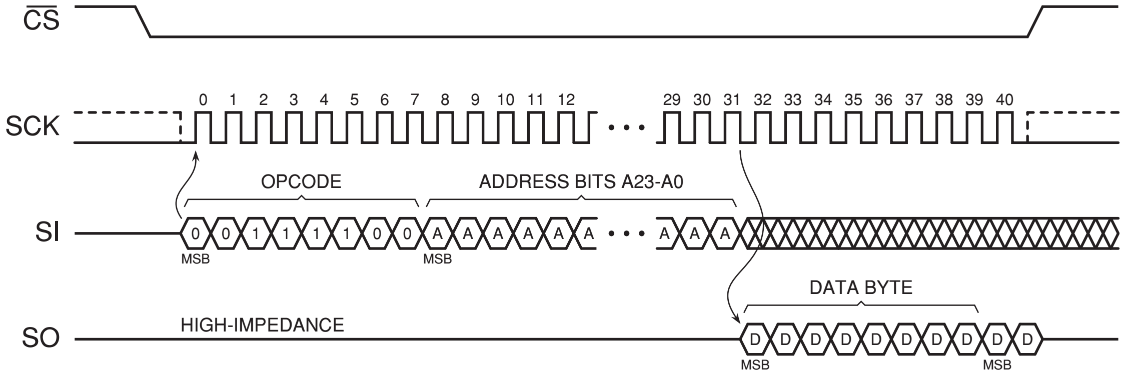
Table 9-3. Read Sector Protection Register – Output Data

Output Data	Sector Protection Register Value
00h	Sector Protection Register value is 0 (sector is unprotected)
FFh	Sector Protection Register value is 1 (sector is protected)

Deasserting the $\overline{\text{CS}}$ pin will terminate the read operation and put the SO pin into a high-impedance state. The $\overline{\text{CS}}$ pin can be deasserted at any time and does not require that a full byte of data be read.

In addition to reading the individual Sector Protection Registers, the Software Protection Status (SWP) bits in the Status Register can be read to determine if all, some, or none of the sectors are software protected (refer to "Read Status Register" on page 31 for more details).

Figure 9-5. Read Sector Protection Register



9.7 Protected States and the Write Protect ($\overline{\text{WP}}$) Pin

The $\overline{\text{WP}}$ pin is not linked to the memory array itself and has no direct effect on the protection status or lockdown status of the memory array. Instead, the $\overline{\text{WP}}$ pin, in conjunction with the SPRL (Sector Protection Registers Locked) bit in the Status Register, is used to control the hardware locking mechanism of the device. For hardware locking to be active, two conditions must be met-the $\overline{\text{WP}}$ pin must be asserted and the SPRL bit must be in the logical “1” state.

When hardware locking is active, the Sector Protection Registers are locked and the SPRL bit itself is also locked. Therefore, sectors that are protected will be locked in the protected state, and sectors that are unprotected will be locked in the unprotected state. These states cannot be changed as long as hardware locking is active, so the Protect Sector, Unprotect Sector, and Write Status Register commands will be ignored. In order to modify the protection status of a sector, the $\overline{\text{WP}}$ pin must first be deasserted, and the SPRL bit in the Status Register must be reset back to the logical “0” state using the Write Status Register command. When resetting the SPRL bit back to a logical “0”, it is not possible to perform a Global Protect or Global Unprotect at the same time since the Sector Protection Registers remain soft-locked until after the Write Status Register command has been executed.

If the $\overline{\text{WP}}$ pin is permanently connected to GND, then once the SPRL bit is set to a logical “1”, the only way to reset the bit back to the logical “0” state is to power-cycle the device. This allows a system to power-up with all sectors software protected but not hardware locked. Therefore, sectors can be unprotected and protected as needed and then hardware locked at a later time by simply setting the SPRL bit in the Status Register.

When the $\overline{\text{WP}}$ pin is deasserted, or if the WP pin is permanently connected to V_{CC}, the SPRL bit in the Status Register can still be set to a logical “1” to lock the Sector Protection Registers. This provides a software locking ability to prevent erroneous Protect Sector or Unprotect Sector commands from being processed. When changing the SPRL bit to a logical “1” from a logical “0”, it is also possible to perform a Global Protect or Global Unprotect at the same time by writing the appropriate values into bits five, four, three, and two of the first byte of the Status Register.

Tables 9-4 and 9-5 detail the various protection and locking states of the device.

Table 9-4. Sector Protection Register States

$\overline{\text{WP}}$	Sector Protection Register n[1]	Sector n[1]
X (Don't Care)	0	Unprotected
	1	Protected

Note: 1. “n” represents a sector number

Table 9-5. Hardware and Software Locking

$\overline{\text{WP}}$	SPRL	Locking	SPRL Change Allowed	Sector Protection Registers
0	0		Can be modified from 0 to 1	Unlocked and modifiable using the Protect and Unprotect Sector commands. Global Protect and Unprotect can also be performed.
0	1	Hardware Locked	Locked	Locked in current state. Protect and Unprotect Sector commands will be ignored. Global Protect and Unprotect cannot be performed.
1	0		Can be modified from 0 to 1	Unlocked and modifiable using the Protect and Unprotect Sector commands. Global Protect and Unprotect can also be performed.
1	1	Software Locked	Can be modified from 1 to 0	Locked in current state. Protect and Unprotect Sector commands will be ignored. Global Protect and Unprotect cannot be performed.

10. Security Commands

10.1 Sector Lockdown

Certain applications require that portions of the Flash memory array be permanently protected against malicious attempts at altering program code, data modules, security information, or encryption/decryption algorithms, keys, and routines. To address these applications, the device incorporates a sector lockdown mechanism that allows any combination of individual 64-Kbyte sectors to be permanently locked so that they become read only. Once a sector is locked down, it can never be erased or programmed again, and it can never be unlocked from the locked down state.

Each 64-Kbyte physical sector has a corresponding single-bit Sector Lockdown Register that is used to control the lockdown status of that sector. These registers are nonvolatile and will retain their state even after a device power-cycle or reset operation. The following table outlines the two states of the Sector Lockdown Registers.

Table 10-1. Sector Lockdown Register Values

Value	Sector Lockdown Status
0	Sector is not locked down and can be programmed and erased. This is the default state.
1	Sector is permanently locked down and can never be programmed or erased again.

Issuing the Sector Lockdown command to a particular sector address will set the corresponding Sector Lockdown Register to the logical “1” state. Each Sector Lockdown Register can only be set once; therefore, once set to the logical “1” state, a Sector Lockdown Register cannot be reset back to the logical “0” state.

Before the Sector Lockdown command can be issued, the Write Enable command must have been previously issued to set the WEL bit in the Status Register to a logical “1”. In addition, the Sector Lockdown Enabled (SLE) bit in the Status Register must have also been previously set to the logical “1” state by using the Write Status Register Byte 2 command (see “Write Status Register Byte 2” on page 36). To issue the Sector Lockdown command, the $\overline{CS}$ pin must first be asserted and the opcode of 33h must be clocked into the device followed by three address bytes designating any address within the 64-Kbyte sector to be locked down. After the three address bytes have been clocked in, a confirmation byte of D0h must also be clocked in immediately following the three address bytes. Any additional data clocked into the device after the first byte of data will be ignored. When the $\overline{CS}$ pin is deasserted, the Sector Lockdown Register corresponding to the sector addressed by A23-A0 will be set to the logical “1” state, and the sector itself will then be permanently locked down from program and erase operations within a time of t_{LOCK}. In addition, the WEL bit in the Status Register will be reset back to the logical “0” state.

The complete three address bytes and the correct confirmation byte value of D0h must be clocked into the device before the $\overline{CS}$ pin is deasserted, and the $\overline{CS}$ pin must be deasserted on an even byte boundary (multiples of eight bits); otherwise, the device will abort the operation. When the device aborts the Sector Lockdown operation, the state of the corresponding Sector Lockdown Register as well as the SLE bit in the Status Register will be unchanged; however, the WEL bit in the Status Register will be reset to a logical “0”.

As a safeguard against accidental or erroneous locking down of sectors, the Sector Lockdown command can be enabled and disabled as needed by using the SLE bit in the Status Register. In addition, the current sector lockdown state can be frozen so that no further modifications to the Sector Lockdown Registers can be made (see “Freeze Sector Lockdown State” below). If the Sector Lockdown command is disabled or if the sector lockdown state is frozen, then any attempts to issue the Sector Lockdown command will be ignored, and the device will reset the WEL bit in the Status Register back to a logical “0” and return to the idle state once the $\overline{CS}$ pin has been deasserted.