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

M25PX32 NOR Serial Flash Embedded Memory

32Mb, Dual I/O, 4KB Subsector Erase, 3V Serial Flash Memory with 75 MHz SPI Bus Interface

Features

- SPI bus compatible serial interface
- 75 MHz (maximum) clock frequency
- 2.7V to 3.6V single supply voltage
- Dual input/output commands resulting in an equivalent clock frequency of 150 MHz
  - DUAL OUTPUT FAST READ command
  - DUAL INPUT FAST PROGRAM command
- Continuous READ of entire memory using FAST READ or DUAL OUTPUT FAST READ command
- 32Mb Flash memory
  - Uniform 4KB subsectors
  - Uniform 64KB sectors
- Additional 64-byte user-lockable, one-time programmable (OTP) area
- Erase capability
  - Subsector (4KB granularity)
  - Sector (64KB granularity)
  - Bulk erase (32Mb) in 34 s typical
- Write protections
  - Software write protection: applicable to every 64KB sector (volatile lock bit)
  - Hardware write protection: non-volatile bits BP0, BP1, BP2 define protected area size
- Deep power down: 5µA typical
- Electronic signature
  - JEDEC standard 2-byte signature (7116h)
  - Unique ID code (UID) with 16-byte read-only space, available upon request
- More than 100,000 write cycles per sector
- More than 20 years data retention
- Packages (RoHS compliant)
  - VFQFPN8 (MP) 6mm x 5mm (MLP8)
  - SO8W (MW) 208 mils width
  - SO16 (MF) 300 mils width
  - TBGA24 (ZM) 6mm x 8mm
- Automotive certified parts available
- This latest generation of this product line is available in the N25Q device (see TN 12-13 for migration guide)

Contents

List of Figures

List of Tables

Functional Description

The M25PX32 is a 32Mb (4Mb x 8) serial Flash memory, with advanced write protection mechanisms, accessed by a high speed SPI-compatible bus. The device supports two high-performance dual input/output instructions that double the transfer bandwidth for read and program operations:

- DUAL OUTPUT FAST READ instruction reads data at up to 75 MHz by using both pin DQ1 and pin DQ0 as outputs.
- DUAL INPUT FAST PROGRAM instruction programs data at up to 75 MHz by using both pin DQ1 and pin DQ0 as inputs.

The memory can be programmed 1 to 256 bytes at a time, using the PAGE PROGRAM instruction. It is organized as 64 sectors that are further divided into 16 subsectors each (1024 total subsectors).

The memory can be erased a 4KB subsector at a time, a 64KB sector at a time, or as a whole. It can be write protected by software using a mix of volatile and non-volatile protection features, depending on the application needs. Protection granularity is 64KB sectors.

The device has 64 one-time-programmable bytes (OTP bytes) that can be read and programmed using two dedicated instructions, READ OTP and PROGRAM OTP, respectively. These 64 bytes can be locked permanently using a PROGRAM OTP command sequence.

Further features are available as additional security options. More information on these security features is available, upon completion of an NDA (nondisclosure agreement), and are, therefore, not described in this datasheet. For more details of this option contact your nearest Micron sales office.

Figure 1: Logic Diagram

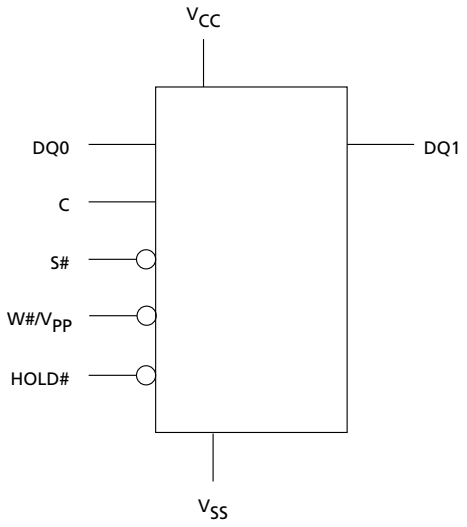
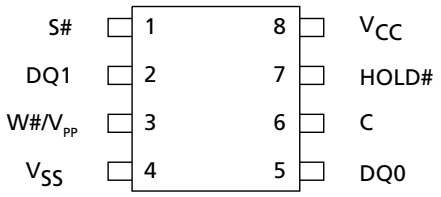


Table 1: Signal Names

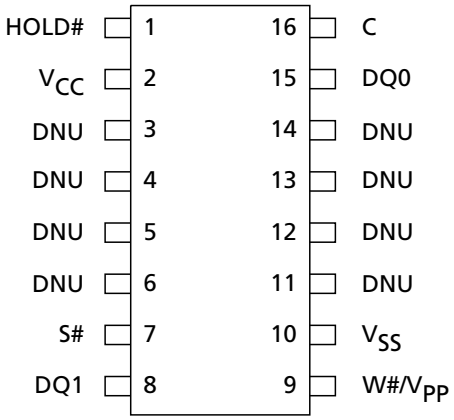
Signal Name	Function	Direction
C	Serial clock	Input
DQ0	Serial data input (Serves as output during DUAL OUTPUT FAST READ operation)	I/O
DQ1	Serial data output (Serves as input during DUAL INPUT FAST PROGRAM operation)	I/O
S#	Chip select	Input
W#/V_{PP}	Write protect or enhanced program supply voltage	Input
HOLD#	Hold	Input
V_{CC}	Supply voltage	
V_{SS}	Ground	

Figure 2: Pin Connections: VFQFPN and SO8



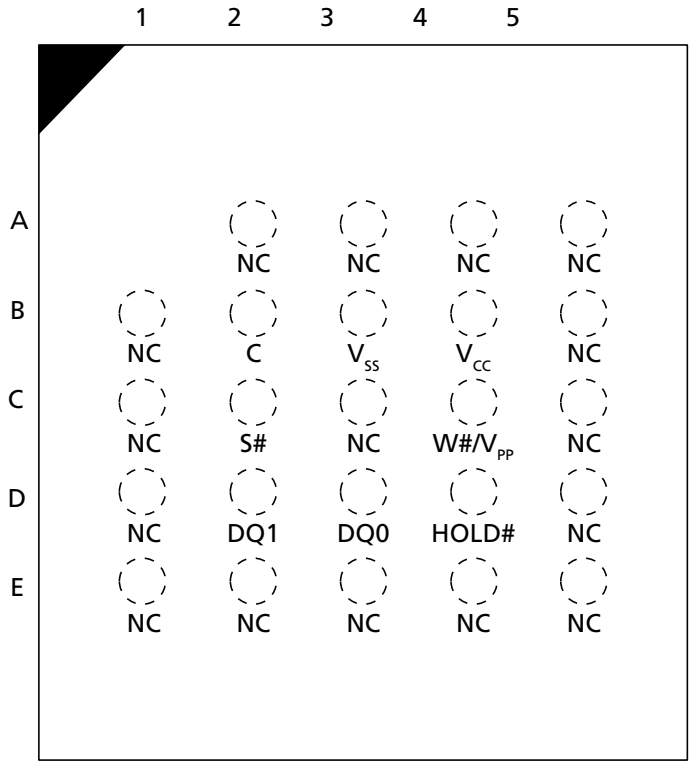
There is an exposed central pad on the underside of the VFQFPN package. This is pulled internally to V_{SS}, and must not be connected to any other voltage or signal line on the PCB. The Package Mechanical section provides information on package dimensions and how to identify pin 1.

Figure 3: Pin Connections: SO16



Note: 1. DNU = do not use.

Figure 4: Pinout: 24-Ball BGA, 6x8mm



Note: 1. DNU = do not use. NC = no connect.

Signal Descriptions

Table 2: Signal Descriptions

Signal	Type	Description
DQ1	Output	**Serial data:** The DQ1 output signal is used to transfer data serially out of the device. Data is shifted out on the falling edge of the serial clock (C). During the DUAL INPUT FAST PROGRAM command, pin DQ1 is used as an input. It is latched on the rising edge of C.
DQ0	Input	**Serial data:** The DQ0 input signal is used to transfer data serially into the device. It receives commands, addresses, and the data to be programmed. Values are latched on the rising edge of the serial clock (C). During the DUAL OUTPUT FAST READ command, pin DQ0 is used as an output. Data is shifted out on the falling edge of C.
C	Input	**Clock:** The C input signal provides the timing of the serial interface. Commands, addresses, or data present at serial data input (DQ0) is latched on the rising edge of the serial clock (C). Data on DQ1 changes after the falling edge of C.
S#	Input	**Chip select:** When the S# input signal is HIGH, the device is deselected and DQ1 is at HIGH impedance. Unless an internal PROGRAM, ERASE, or WRITE STATUS REGISTER cycle is in progress, the device will be in the standby power mode (not the DEEP POWER-DOWN mode). Driving S# LOW enables the device, placing it in the active power mode. After power-up, a falling edge on S# is required prior to the start of any command.
HOLD#	Input	**Hold:** The HOLD# signal is used to pause any serial communications with the device without deselecting the device. During the hold condition, DQ1 is High-Z. DQ0 and C are "Don't Care." To start the hold condition, the device must be selected, with S# driven LOW.
W#/V_{PP}	Input	**Write protect/enhanced program supply voltage:**The W#/V_{PP} signal is both a control input and a power supply pin. The two functions are selected by the voltage range applied to the pin. If the W#/V_{PP} input is kept in a low voltage range (0 V to V_{CC}) the pin is seen as a control input. The W# input signal is used to freeze the size of the area of memory that is protected against program or erase commands as specified by the values in BP2, BP1, and BP0 bits of the Status Register. V_{PP} acts as an additional power supply if it is in the range of V_{PPH}, as defined in the AC Measurement Conditions table. Avoid applying V_{PPH} to the W#/V_{PP} pin during a Bulk Erase operation.
V_{CC}	Power	**Device core power supply:** Source voltage.
V_{SS}	Ground	**Ground:** Reference for the V_{CC} supply voltage.

Serial Peripheral Interface Modes

The device can be driven by a microcontroller while its serial peripheral interface (SPI) is in either of the two modes shown here. The difference between the two modes is the clock polarity when the bus master is in standby mode and not transferring data. Input data is latched in on the rising edge of the clock, and output data is available from the falling edge of the clock.

Table 3: SPI Modes

SPI Modes	Clock Polarity
CPOL = 0, CPHA = 0	C remains at 0 for (CPOL = 0, CPHA = 0)
CPOL = 1, CPHA = 1	C remains at 1 for (CPOL = 1, CPHA = 1)

The following figure is an example of three memory devices in a simple connection to an MCU on an SPI bus. Because only one device is selected at a time, that one device drives DQ1, while the other devices are HIGH-Z.

Resistors ensure the device is not selected if the bus master leaves S# HIGH-Z. The bus master might enter a state in which all input/output is HIGH-Z simultaneously, such as when the bus master is reset. Therefore, the serial clock must be connected to an external pull-down resistor so that S# is pulled HIGH while the serial clock is pulled LOW. This ensures that S# and the serial clock are not HIGH simultaneously and that ^{t}SHCH is met. The typical resistor value of 100kΩ, assuming that the time constant R × Cp (Cp = parasitic capacitance of the bus line), is shorter than the time the bus master leaves the SPI bus in HIGH-Z.

Example: Cp = 50 pF, that is R × Cp = 5µs. The application must ensure that the bus master never leaves the SPI bus HIGH-Z for a time period shorter than 5µs. W# and HOLD# should be driven either HIGH or LOW, as appropriate.

Figure 5: Bus Master and Memory Devices on the SPI Bus

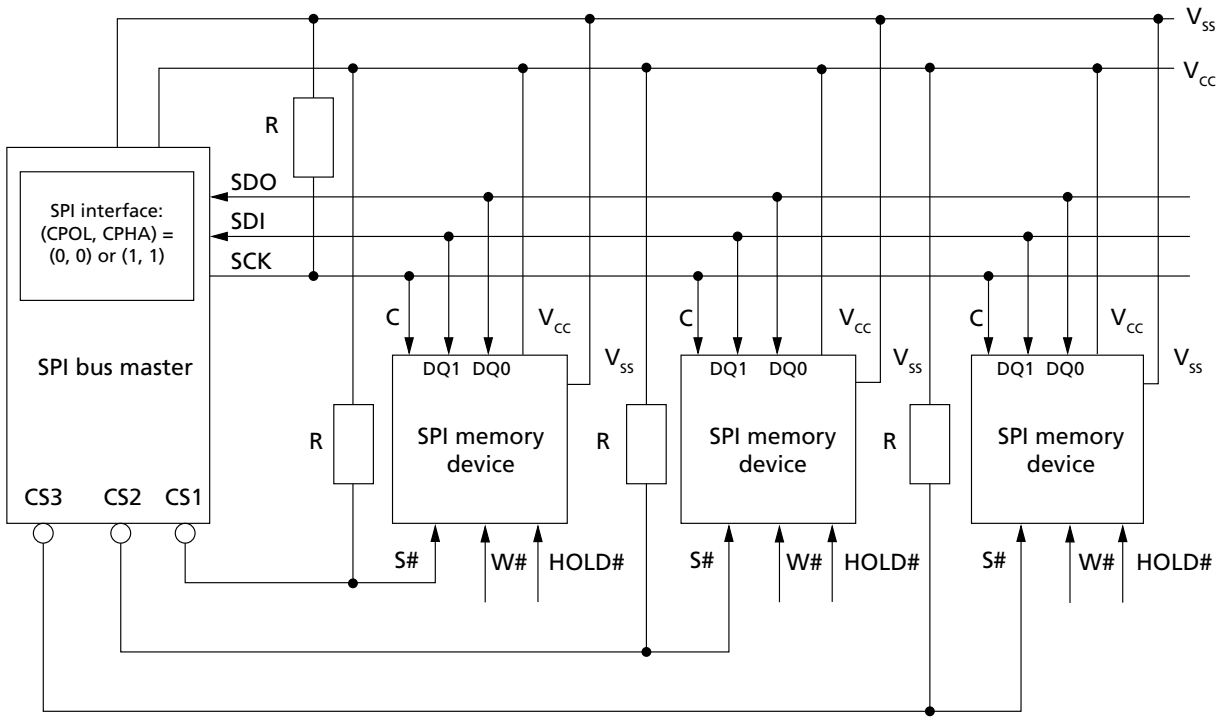
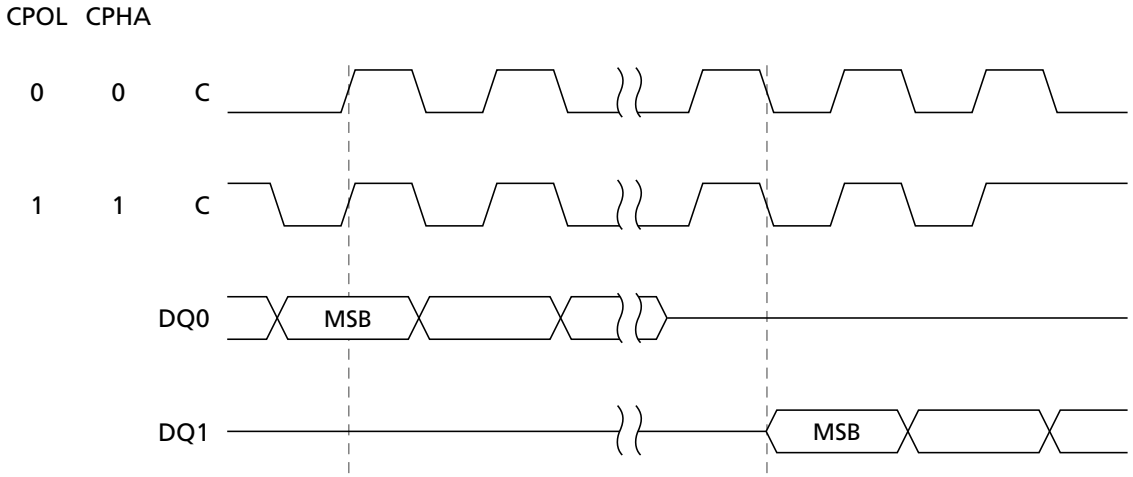


Figure 6: SPI Modes



Operating Features

Page Programming

To program one data byte, two commands are required: WRITE ENABLE, which is one byte, and a PAGE PROGRAM sequence, which is four bytes plus data. This is followed by the internal PROGRAM cycle of duration t_{PP}. To spread this overhead, the PAGE PROGRAM command allows up to 256 bytes to be programmed at a time (changing bits from 1 to 0), provided they lie in consecutive addresses on the same page of memory. To optimize timings, it is recommended to use the PAGE PROGRAM command to program all consecutive targeted bytes in a single sequence than to use several PAGE PROGRAM sequences with each containing only a few bytes.

Dual Input Fast Program

The DUAL INPUT FAST PROGRAM command makes it possible to program up to 256 bytes using two input pins at the same time (by changing bits from 1 to 0). For optimized timings, it is recommended to use the DUAL INPUT FAST PROGRAM command to program all consecutive targeted bytes in a single sequence than to use several DUAL INPUT FAST PROGRAM sequences each containing only a few bytes.

Subsector Erase, Sector Erase, Bulk Erase

The PAGE PROGRAM command allows bits to be reset from 1 to 0. Before this can be applied, the bytes of memory need to have been erased to all 1s (FFh). This can be achieved a subsector at a time using the SUBSECTOR ERASE command, a sector at a time using the SECTOR ERASE command, or throughout the entire memory using the BULK ERASE command. This starts an internal ERASE cycle of duration t_{SSE}, t_{SE} or t_{BE}. The ERASE command must be preceded by a WRITE ENABLE command.

Polling during a Write, Program, or Erase Cycle

An improvement in the time to complete the following commands can be achieved by not waiting for the worst case delay (t_W, t_{PP}, t_{SSE}, t_{SE}, or t_{BE}).

- WRITE STATUS REGISTER
- PROGRAM OTP
- PROGRAM
- DUAL INPUT FAST PROGRAM
- ERASE (SUBSECTOR ERASE, SECTOR ERASE, BULK ERASE)

The write in progress (WIP) bit is provided in the status register so that the application program can monitor this bit in the status register, polling it to establish when the previous WRITE cycle, PROGRAM cycle, or ERASE cycle is complete.

Active Power, Standby Power, and Deep Power-Down

When chip select (S#) is LOW, the device is selected, and in the ACTIVE POWER mode. When S# is HIGH, the device is deselected, but could remain in the ACTIVE POWER mode until all internal cycles have completed (PROGRAM, ERASE, WRITE STATUS REGISTER). The device then goes in to the STANDBY POWER mode. The device consumption drops to I_{CC1}.

The DEEP POWER-DOWN mode is entered when the DEEP POWER-DOWN command is executed. The device consumption drops further to I_{CC2}. The device remains in this mode until the RELEASE FROM DEEP POWER-DOWN command is executed. While in the DEEP POWER-DOWN mode, the device ignores all WRITE, PROGRAM, and ERASE commands. This provides an extra software protection mechanism when the device is not in active use, by protecting the device from inadvertent WRITE, PROGRAM, or ERASE operations. For further information, see DEEP POWER-DOWN (page 44).

Status Register

The status register contains a number of status and control bits that can be read or set (as appropriate) by specific commands. For a detailed description of the status register bits, see (page 0).

Data Protection by Protocol

Non-volatile memory is used in environments that can include excessive noise. The following capabilities help protect data in these noisy environments.

Power on reset and an internal timer (t_{PUW}) can provide protection against inadvertent changes while the power supply is outside the operating specification.

WRITE, PROGRAM, ERASE, and WRITE STATUS REGISTER commands are checked before they are accepted for execution to ensure they consist of a number of clock pulses that is a multiple of eight.

All commands that modify data must be preceded by a WRITE ENABLE command to set the write enable latch (WEL) bit.

In addition to the low power consumption feature, the DEEP POWER-DOWN mode offers extra software protection since all WRITE, PROGRAM, and ERASE commands are ignored when the device is in this mode.

Software Data Protection

Memory can be configured as read-only using the top/bottom bit and the block protect bits (BP2, BP1, BP0) as shown in the Protected Area Sizes table.

Memory sectors can be protected by specific lock registers assigned to each 64KB sector. These lock registers can be read and written using the READ LOCK REGISTER and WRITE to LOCK REGISTER commands. In each lock register the following two bits control the protection of each sector:

- Write lock bit: This bit determines whether the contents of the sector can be modified using the WRITE, PROGRAM, and ERASE commands. When the bit is set to ‘1’, the sector is write protected, and any operations that attempt to change the data in the sector will fail. When the bit is reset to ‘0’, the sector is not write protected by the lock register, and may be modified.
- Lock down bit: This bit provides a mechanism for protecting software data from simple hacking and malicious attack. When the bit is set to '1’, further modification to the write lock bit and lock down bit cannot be performed. A power-up, is required before changes to these bits can be made. When the bit is reset to ‘0’, the write lock bit and lock down bit can be changed.

The software protection truth table shows the lock down bit and write lock bit settings and the sector protection status.

Table 4: Software Protection Truth Table

Sector Lock Register Bits		
Lock Down	**Write Lock**	**Protection Status**
0	0	Sector unprotected from PROGRAM / ERASE / WRITE operations; protection status reversible
0	1	Sector protected from PROGRAM / ERASE / WRITE operations; protection status reversible
1	0	Sector unprotected from PROGRAM / ERASE / WRITE operations; protection status cannot be changed except by a power-up.
1	1	Sector protected from PROGRAM / ERASE / WRITE operations; protection status cannot be changed except by a power-up.

Hardware Data Protection

Hardware data protection is implemented using the write protect signal applied on the W#/V_{PP} pin. This freezes the status register in a read-only mode, protecting the block protect (BP) bits and the status register write disable bit (SRWD). The device is ready to accept a BULK ERASE command only if all block protect bits are 0.

Table 5: Sectors 0 to 16, Protected Area Sizes – Upper Area Protection

Status Register Content				Memory Content	
Top/Bottom Bit	**BP2**	**BP1**	**BP0**	**Protected Area**	**Unprotected Area**
0	0	0	0	None	All sectors [1]
0	0	0	1	Upper 16th (sector 15)	Lower 15/16ths (sectors 0 to 14)
0	0	1	0	Upper 8th (sectors 14 to 15)	Lower 7/8ths (sectors 0 to 13)
0	0	1	1	Upper 4th (sectors 12 to 15)	Lower 3/4ths (sectors 0 to 11)
0	1	0	0	Upper half (sectors 8 to 15)	Lower half (sectors 0 to 7)
0	1	0	1	All sectors	None
0	1	1	0	All sectors	None
0	1	1	1	All sectors	None

Note: 1. The device is ready to accept a BULK ERASE command only if all block protect bits are 0.

Table 6: Sectors 0 to 16, Protected Area Sizes – Lower Area Protection

Status Register Content				Memory Content	
Top/Bottom Bit	**BP2**	**BP1**	**BP0**	**Protected Area**	**Unprotected Area**
1	0	0	0	None	All sectors [1]
1	0	0	1	Lower 16th (sector 0)	Upper 15/16ths (sectors 1 to 15)
1	0	1	0	Lower 8th (sectors 0 to 1)	Upper 7/8ths (sectors 2 to 15)
1	0	1	1	Lower 4th (sectors 0 to 3)	Upper 3/4ths (sectors 4 to 15)
1	1	0	0	Lower half (sectors 3 to 7)	Upper half (sectors 8 to 15)
1	1	0	1	All sectors	None
1	1	1	0	All sectors	None

Table 6: Sectors 0 to 16, Protected Area Sizes – Lower Area Protection (Continued)

Status Register Content				Memory Content	
Top/Bottom Bit	BP2	BP1	BP0	Protected Area	Unprotected Area
1	1	1	1	All sectors	None

Note: 1. The device is ready to accept a BULK ERASE command only if all block protect bits are 0.

Table 7: Sectors 0 to 32, Protected Area Sizes – Upper Area Protection

Status Register Content				Memory Content	
Top/Bottom Bit	BP 2	BP 1	BP 0	Protected Area	Unprotected Area
0	0	0	0	none	All sectors [1]
0	0	0	1	Upper 32nd (sector 31)	Lower 31/32nds (sectors 0 to 30)
0	0	1	0	Upper 16th (sectors 30 to 31)	Lower 15/16ths (sectors 0 to 29)
0	0	1	1	Upper 8th (sectors 28 to 31)	Lower 7/8ths (sectors 0 to 27)
0	1	0	0	Upper 4th (sectors 24 to 31)	Lower 3/4ths (sectors 0 to 23)
0	1	0	1	Upper half (sectors 16 to 31)	Lower half (sectors 0 to 15)
0	1	1	0	All sectors	none
0	1	1	1	All sectors	none

Note: 1. The device is ready to accept a BULK ERASE command only if all block protect bits are 0.

Table 8: Sectors 0 to 32, Protected Area Sizes – Lower Area Protection

Status Register Content				Memory Content	
Top/Bottom Bit	BP 2	BP 1	BP 0	Protected Area	Unprotected Area
1	0	0	0	none	All sectors [1]
1	0	0	1	Lower 32nd (sector 0)	Upper 31/32nds (sectors 1 to 31)
1	0	1	0	Lower 16th (sectors 0 to 1)	Upper 15/16ths (sectors 2 to 31)
1	0	1	1	Lower 8th (sectors 0 to 3)	Upper 7/8ths (sectors 4 to 31)
1	1	0	0	Lower 4th (sectors 0 to 7)	Upper 3/4ths (sectors 8 to 31)
1	1	0	1	Lower half (sectors 0 to 15)	Upper half (sectors 16 to 31)
1	1	1	0	All sectors	none
1	1	1	1	All sectors	none

Note: 1. The device is ready to accept a BULK ERASE command only if all block protect bits are 0.

Table 9: Sectors 0 to 63, Protected Area Sizes – Upper Area Protection

Status Register Content				Memory Content	
Top/Bottom Bit	BP2	BP1	BP0	Protected Area	Unprotected Area
0	0	0	0	None	All sectors [1]
0	0	0	1	Upper 64th (sector 63)	Lower 63/64ths (sectors 0 to 62)
0	0	1	0	Upper 32th (sectors 62 to 63)	Lower 31/32nds (sectors 0 to 61)

Table 9: Sectors 0 to 63, Protected Area Sizes – Upper Area Protection (Continued)

Status Register Content				Memory Content	
Top/Bottom Bit	BP2	BP1	BP0	Protected Area	Unprotected Area
0	0	1	1	Upper 16th (sectors 60 to 63)	Lower 15/16ths (sectors 0 to 59)
0	1	0	0	Upper 8th (sectors 56 to 63)	Lower 7/8ths (sectors 0 to 55)
0	1	0	1	Upper 4th (sectors 48 to 63)	Lower 3/4ths (sectors 0 to 47)
0	1	1	0	Upper half (sectors 32 to 63)	Lower half (sectors 0 to 31)
0	1	1	1	All sectors	None

Note: 1. The device is ready to accept a BULK ERASE command only if all block protect bits are 0.

Table 10: Sectors 0 to 63, Protected Area Sizes – Lower Area Protection

Status Register Content				Memory Content	
Top/Bottom Bit	BP2	BP1	BP0	Protected Area	Unprotected Area
1	0	0	0	None	All sectors [1]
1	0	0	1	Lower 64th (sector 0)	Upper 63/64ths (sectors 1 to 63)
1	0	1	0	Lower 32th (sectors 0 to 1)	Upper 31/32nds (sectors 2 to 63)
1	0	1	1	Lower 16th (sectors 0 to 3)	Upper 15/16ths (sectors 4 to 63)
1	1	0	0	Lower 8th (sectors 0 to 7)	Upper 7/8ths (sectors 8 to 63)
1	1	0	1	Lower 4th (sectors 0 to 15)	Upper 3/4ths (sectors 16 to 63)
1	1	1	0	Lower half (sectors 0 to 31)	Upper half (sectors 32 to 63)
1	1	1	1	All sectors	None

Note: 1. The device is ready to accept a BULK ERASE command only if all block protect bits are 0.

Table 11: Sectors 0 to 127, Protected Area Sizes – Upper Area Protection

Status Register Content				Memory Content	
Top/Bottom Bit	BP2	BP1	BP0	Protected Area	Unprotected Area
0	0	0	0	None	All sectors [1]
0	0	0	1	Upper 64th (sectors 126 to 127)	Lower 63/64ths (sectors 0 to 125)
0	0	1	0	Upper 32nd (sectors 124 to 127)	Lower 31/32nds (sectors 0 to 123)
0	0	1	1	Upper 16th (sectors 120 to 127)	Lower 15/16ths (sectors 0 to119)
0	1	0	0	Upper 8th (sectors 112 to 127)	Lower 7/8ths (sectors 0 to 111)
0	1	0	1	Upper 4th (sectors 96 to 127)	Lower 3/4ths (sectors 0 to 95)
0	1	1	0	Upper half (sectors 64 to 127)	Lower half (sectors 0 to 63)
0	1	1	1	All sectors	None

Note: 1. The device is ready to accept a BULK ERASE command only if all block protect bits are 0.

Table 12: Sectors 0 to 127, Protected Area Sizes – Lower Area Protection

Status Register Content				Memory Content	
Top/Bottom Bit	BP2	BP1	BP0	Protected Area	Unprotected Area
1	0	0	0	None	All sectors [1]
1	0	0	1	Lower 64th (sectors 0 to 1)	Upper (sectors 2 to 127)
1	0	1	0	Lower 32nd (sectors 0 to 3)	Upper (sectors 4 to 127)
1	0	1	1	Lower 16th (sectors 0 to 7)	Upper (sectors 8 to 127)
1	1	0	0	Lower 8th (sectors 0 to 15)	Upper (sectors 16 to 127)
1	1	0	1	Lower 4th (sectors 0 to 31)	Upper (sectors 32 to 127)
1	1	1	0	Lower half (sectors 0 to 63)	Upper (sectors 64 to 127)
1	1	1	1	None	All sectors

Note: 1. The device is ready to accept a BULK ERASE command only if all block protect bits are 0.

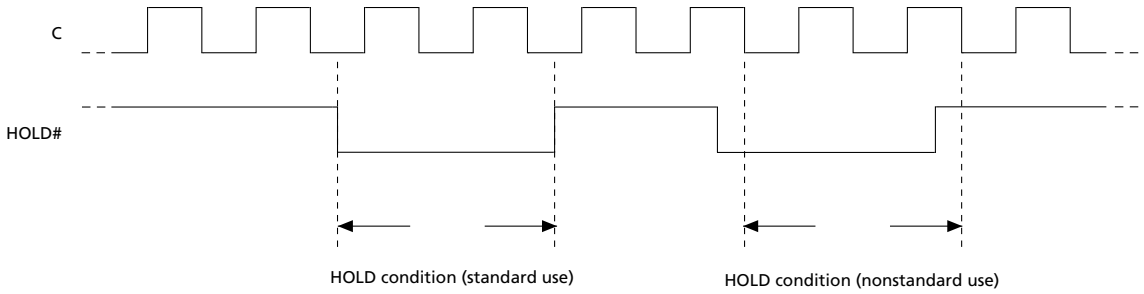
Hold Condition

The HOLD# signal is used to pause any serial communications with the device without resetting the clocking sequence. However, taking this signal LOW does not terminate any WRITE STATUS REGISTER, PROGRAM, or ERASE cycle that is currently in progress.

To enter the hold condition, the device must be selected, with S# LOW. The hold condition starts on the falling edge of the HOLD# signal, if this coincides with serial clock (C) being LOW. The hold condition ends on the rising edge of the HOLD# signal, if this coincides with C being LOW. If the falling edge does not coincide with C being LOW, the hold condition starts after C next goes LOW. Similarly, if the rising edge does not coincide with C being LOW, the hold condition ends after C next goes LOW.

During the hold condition, DQ1 is HIGH impedance while DQ0 and C are Don't Care. Typically, the device remains selected with S# driven LOW for the duration of the hold condition. This ensures that the state of the internal logic remains unchanged from the moment of entering the hold condition. If S# goes HIGH while the device is in the hold condition, the internal logic of the device is reset. To restart communication with the device, it is necessary to drive HOLD# HIGH, and then to drive S# LOW. This prevents the device from going back to the hold condition.

Figure 7: Hold Condition Activation

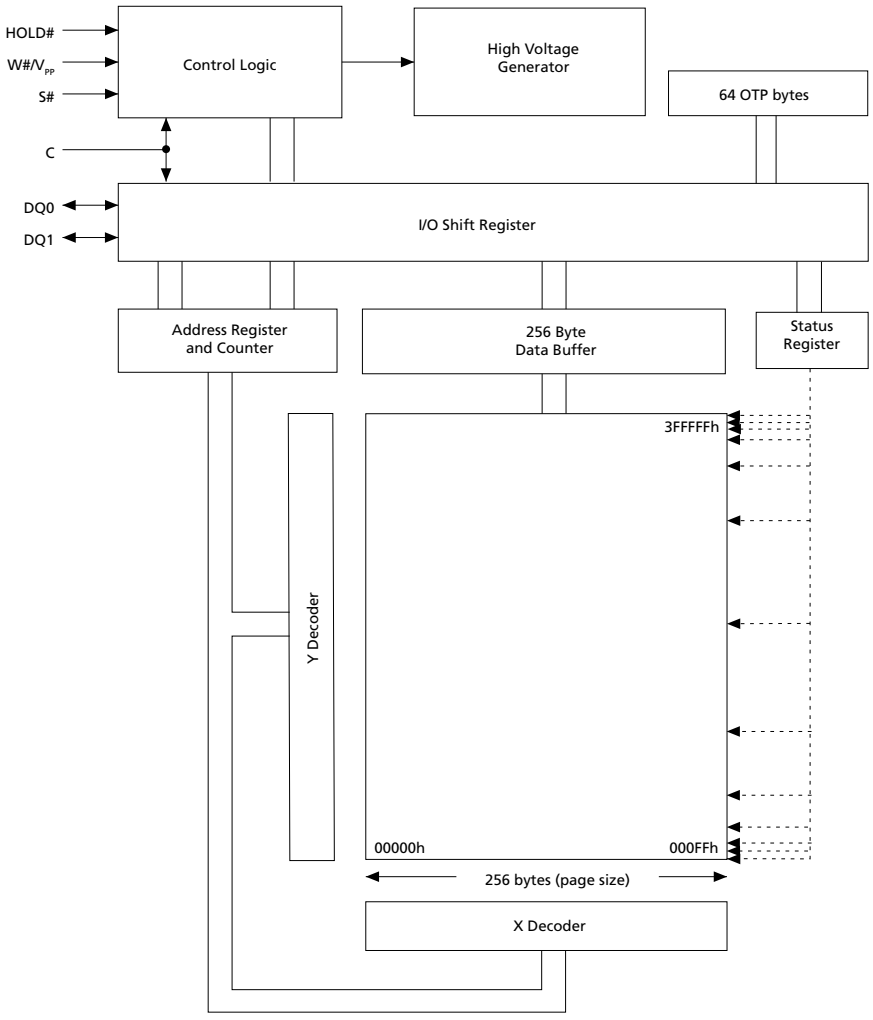


Memory Configuration and Block Diagram

Each page of memory can be individually programmed; bits are programmed from 1 to 0. The device is sector or bulk-erasable, but not page-erasable; bits are erased from 0 to 1. The memory is configured as follows:

- 4,194,304 bytes (8 bits each)
- 1024 subsectors (4KB each)
- 64 sectors (64KB each)
- 16,384 pages (256 bytes each)
- 64 OTP bytes located outside main memory

Figure 8: Block Diagram



Memory Map – 32Mb Density

Table 13: Sectors[127:0]

Sector	Subsector	Address Range	
		Start	End
63	1023	003F F000h	003F FFFFh
	⋮	⋮	⋮
	1008	003F 0000h	003F 0FFFh
⋮	⋮	⋮	⋮
0	15	0000 F000h	0000 FFFFh
	⋮	⋮	⋮
	0	0000 0000h	0000 0FFFh

Command Set Overview

All commands, addresses, and data are shifted in and out of the device, most significant bit first.

Serial data inputs DQ0 and DQ1 are sampled on the first rising edge of serial clock (C) after chip select (S#) is driven LOW. Then, the one-byte command code must be shifted in to the device, most significant bit first, on DQ0 and DQ1, each bit being latched on the rising edges of C.

Every command sequence starts with a one-byte command code. Depending on the command, this command code might be followed by address or data bytes, by address and data bytes, or by neither address or data bytes. For the following commands, the shifted-in command sequence is followed by a data-out sequence. S# can be driven HIGH after any bit of the data-out sequence is being shifted out.

- READ DATA BYTES (READ)
- READ DATA BYTES at HIGHER SPEED
- DUAL OUTPUT FAST READ
- READ OTP
- READ LOCK REGISTERS
- READ STATUS REGISTER
- READ IDENTIFICATION
- RELEASE from DEEP POWER-DOWN

For the following commands, S# must be driven HIGH exactly at a byte boundary. That is, after an exact multiple of eight clock pulses following S# being driven LOW, S# must be driven HIGH. Otherwise, the command is rejected and not executed.

- PAGE PROGRAM
- PROGRAM OTP
- DUAL INPUT FAST PROGRAM
- SUBSECTOR ERASE
- SECTOR ERASE
- BULK ERASE
- WRITE STATUS REGISTER
- WRITE to LOCK REGISTER
- WRITE ENABLE
- WRITE DISABLE
- DEEP POWER-DOWN

All attempts to access the memory array are ignored during a WRITE STATUS REGISTER command cycle, a PROGRAM command cycle, or an ERASE command cycle. In addition, the internal cycle for each of these commands continues unaffected.

Table 14: Command Set Codes

Command Name	One-Byte Command Code		Bytes		
			Address	Dummy	Data
WRITE ENABLE	0000 0110	06h	0	0	0
WRITE DISABLE	0000 0100	04h	0	0	0
READ IDENTIFICATION	1001 1111	9Fh	0	0	1 to 20
	1001 1110	9Eh			1 to 20
READ STATUS REGISTER	0000 0101	05h	0	0	1 to ∞
WRITE STATUS REGISTER	0000 0001	01h	0	0	1
WRITE to LOCK REGISTER	1110 0101	E5h	3	0	1
READ LOCK REGISTER	1110 1000	E8h	3	0	1
READ DATA BYTES	0000 0011	03h	3	0	1 to ∞
READ DATA BYTES at HIGHER SPEED	0000 1011	0Bh	3	1	1 to ∞
DUAL OUTPUT FAST READ	0011 1011	3Bh	3	1	1 to ∞
READ OTP (Read 64 bytes of OTP area)	0100 1011	4Bh	3	1	1 to 65
PROGRAM OTP (Program 64 bytes of OTP area)	0100 0010	42h	3	0	1 to 65
PAGE PROGRAM	0000 0010	02h	3	0	1 to 256
DUAL INPUT FAST PROGRAM	1010 0010	A2h	3	0	1 to 256
SUBSECTOR ERASE	0010 0000	20h	3	0	0
SECTOR ERASE	1101 1000	D8h	3	0	0
BULK ERASE	1100 0111	C7h	0	0	0
DEEP POWER-DOWN	1011 1001	B9h	0	0	0
RELEASE from DEEP POWER-DOWN	1010 1011	ABh	0	0	0

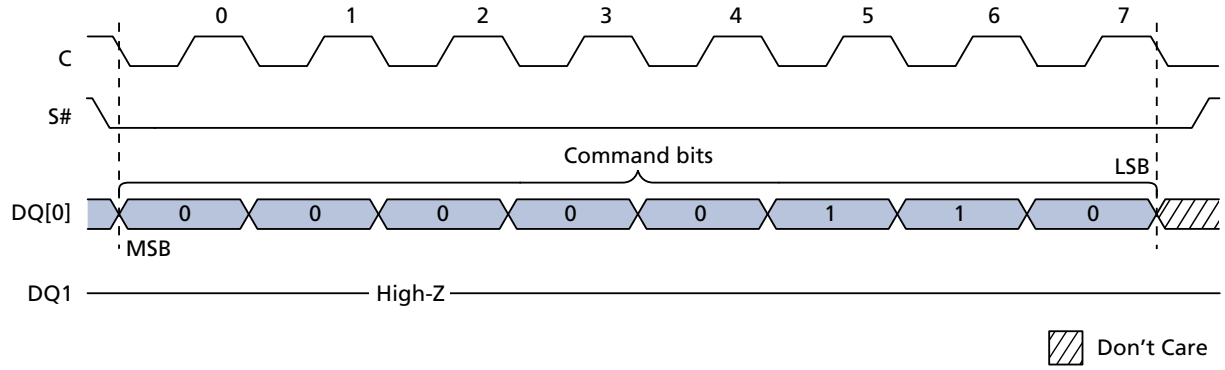
WRITE ENABLE

The WRITE ENABLE command sets the write enable latch (WEL) bit.

The WEL bit must be set before execution of every PROGRAM, ERASE, and WRITE command.

The WRITE ENABLE command is entered by driving chip select (S#) LOW, sending the command code, and then driving S# HIGH.

Figure 9: WRITE ENABLE Command Sequence



WRITE DISABLE

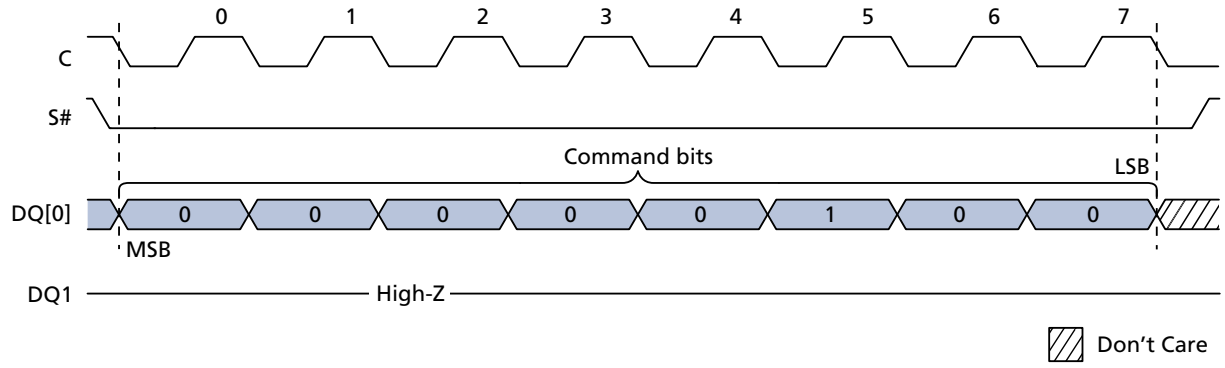
The WRITE DISABLE command resets the write enable latch (WEL) bit.

The WRITE DISABLE command is entered by driving chip select (S#) LOW, sending the command code, and then driving S# HIGH.

The WEL bit is reset under the following conditions:

- Power-up
- Completion of any ERASE operation
- Completion of any PROGRAM operation
- Completion of any WRITE REGISTER operation
- Completion of WRITE DISABLE operation

Figure 10: WRITE DISABLE Command Sequence



READ IDENTIFICATION

The READ IDENTIFICATION command reads the following device identification data:

- Manufacturer identification (1 byte): This is assigned by JEDEC.
- Device identification (2 bytes): This is assigned by device manufacturer; the first byte indicates memory type and the second byte indicates device memory capacity.
- A Unique ID code (UID) (17 bytes,16 available upon customer request): The first byte contains length of data to follow; the remaining 16 bytes contain optional Customized Factory Data (CFD) content.

Table 15: READ IDENTIFICATION Data Out Sequence

Manufacturer Identification	Device Identification		UID	
	Memory Type	Memory Capacity	CFD Length	CFD Content
20h	20h	11h	10h	16 bytes
	71h	12h		
		13h		
		14h		
		15h		
		16h		
		17h		
		18h		

Note: 1. The CFD bytes are read-only and can be programmed with customer data upon demand. If customers do not make requests, the devices are shipped with all the CFD bytes programmed to zero.

A READ IDENTIFICATION command is not decoded while an ERASE or PROGRAM cycle is in progress and has no effect on a cycle in progress. The READ IDENTIFICATION command must not be issued while the device is in DEEP POWER-DOWN mode.

The device is first selected by driving S# LOW. Then the 8-bit command code is shifted in and content is shifted out on DQ1 as follows: the 24-bit device identification that is stored in the memory, the 8-bit CFD length, followed by 16 bytes of CFD content. Each bit is shifted out during the falling edge of serial clock (C).

The READ IDENTIFICATION command is terminated by driving S# HIGH at any time during data output. When S# is driven HIGH, the device is put in the STANDBY POWER mode and waits to be selected so that it can receive, decode, and execute commands.

Figure 11: READ IDENTIFICATION Command Sequence