

# A Guide to SPI F-RAM Devices

*Covers Functional Description, Timing, Pseudo Code*



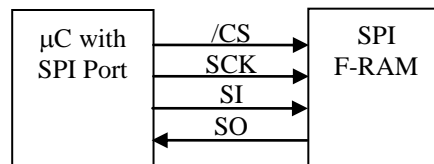
## OVERVIEW

The SPI family of F-RAM products offers the market serial high speed, low power, nonvolatile F-RAM memory. SPI F-RAM memory devices take advantage of F-RAM's fast write performance that provides writing at bus speeds up to 40 MHz without waiting for writes to complete. SPI F-RAM densities start at 4 Kb (512 Byte) and extend to 2 Mb (256 KByte).

These devices employ an industry standard 4-wire SPI interface. The interface is synchronous. A clock SCK is used to latch data into and out of the device, and a chip select pin is used to enable/disable the device. Stand-alone SPI F-RAM devices are designated FM25xxx. The FM24xxx part numbers designate the stand-alone 2-wire (I<sup>2</sup>C) F-RAM family.

## WHY USE SPI?

The Serial Peripheral Interface (SPI) is a serial bus created by Motorola (now Freescale) and is provided as a dedicated interface on their MCUs and those from other semiconductor suppliers, such as TI, Atmel, Microchip, Analog Devices, etc. Other processor types have an SPI port as well, e.g. DSPs and network processors. For microcontroller-based systems that require high serial data rates, the SPI interface is an ideal choice. SPI F-RAM memories offer the highest speed SPI devices in Ramtron's serial memory product line. Some can transfer data up to 40 Mbit/sec with as few as 3 pins. A dedicated SPI port on a microcontroller makes the choice an easy one in those cases. However, many microcontrollers do not have a dedicated SPI port, so the use of bit-banging provides a means to use general purpose I/O pins. This method involves software that controls or "bangs away" at the I/O port.

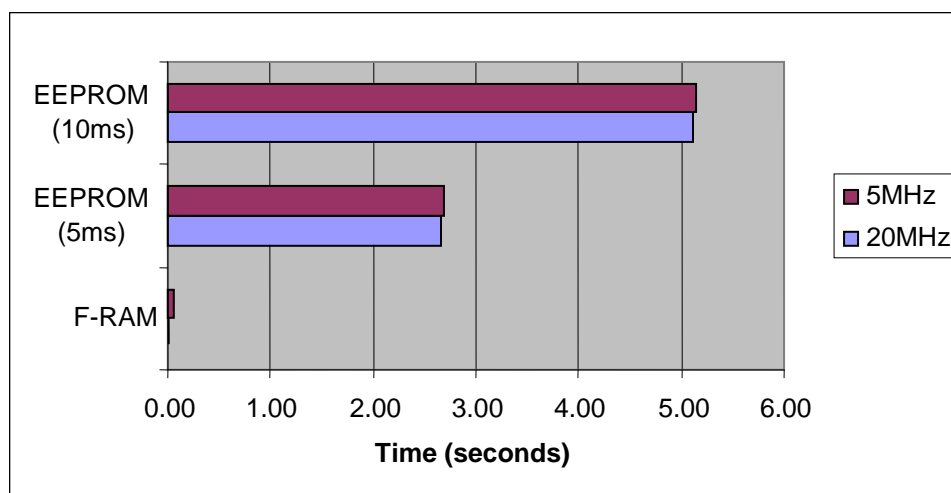


**Figure 1. Basic SPI Interface**

## SPEED ADVANTAGES

The F-RAM memory technology allows a system designer to write large data blocks much faster than EEPROM or Flash equivalents. The SPI interface operates up to 40 MHz but not only clock speed but F-RAM memories do not impose write delays on the system. F-RAM writes each data byte immediately following the 8<sup>th</sup> bit in each byte received. That is, F-RAM devices do not use a page buffer, which is used on all EEPROMs and Serial flash memories. Those devices use it to temporarily store write data, which is then written to the EEPROM/flash array within a 5 to 10 ms period. The combination of no write delays and high clock speed makes the F-RAM compelling for any application that needs to write a lot of data quickly. The system designer has complete freedom over how many bytes to write to the SPI F-RAM. When you write a byte or two in random locations in an F-RAM, the write cycle time is approximately 1μs, whereas an EEPROM/flash imposes its 5 to 10 ms write cycle on you. Also there are no worries about page buffer sizes that change when your system requires the next memory density a few years later.

The chart below shows the time required to write a 256Kb array - F-RAM vs. EEPROM. Even for an EEPROM with a 64-byte page buffer, the F-RAM device is orders of magnitude faster at the same clock rate. This is especially significant on a production line where there is a limited time to program system settings.



**Figure 2. Write Time to Fill a 256Kb SPI Memory Array**

Note that the time to write a 256Kb EEPROM memory is not improved much by increasing the clock frequency from 5 MHz to 20 MHz. The long write delay needed for each page write dominates.

For a 20MHz F-RAM memory, the entire 32K byte array can be written in just 13 ms, which is a value small enough that it does not appear on the above chart.

## THE SPI BUS

All transactions occur with /CS low while address, control, and data are serially clocked into the device in byte-size blocks. Address, control, and data-in are clocked in on the SI pin, and data-out is clocked out on the SO pin. “Op-codes” provide control over the device. Read and write transactions follow the sequence: op-code, address, data. Other transactions that do not involve a data transfer are used to set/clear the WEL bit in the Status Register. The Status Register holds memory write-protect settings. For EEPROM- and flash-based SPI memories, the Status Register also holds an important bit called /RDY. It is the ready flag that tells the SPI controller if a write cycle has completed or not. EEPROMs and flash memories typically require 5 to 10 ms of delay before the device can be accessed after a write. With F-RAM there are no delays, no waiting for the internal write to complete, and therefore the /RDY bit is always a logic ‘0’. The controller can read and write the memory like a true RAM. There are no write delays and no page buffer limitations to constrain the user.

There are six op-codes that control SPI F-RAM devices. The FM25H20 device adds a seventh op-code for Sleep entry. Each op-code is an 8-bit command that instructs the memory or Status Register to do something. There is only one op-code for each active /CS cycle. Table 1 describes all the op-codes and is shown on the following page.

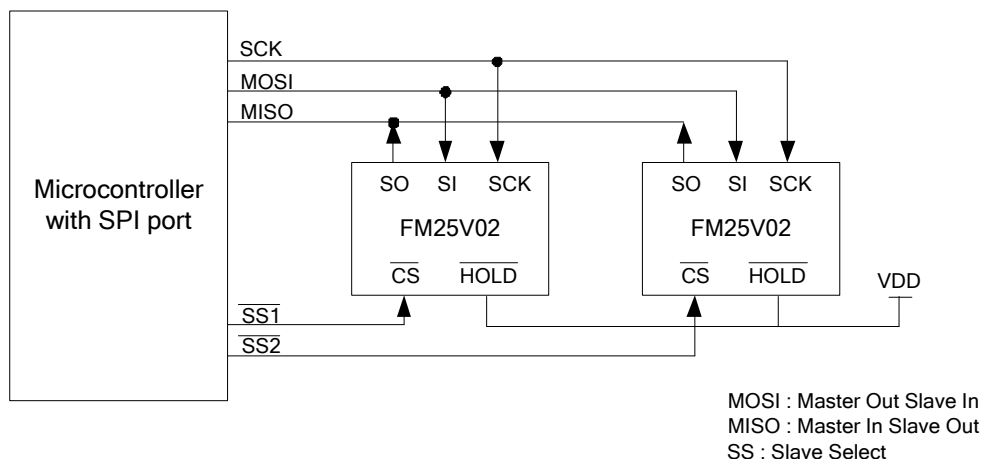
**Table 1. Description of Op-Codes**

Name	Op-Code	Address	Data	Action
WREN	0000_0110b	-	-	Sets WEL
WRITE	0000_0010b	2-byte*	Memory Data in	Writes data to F-RAM array if WEL=1. When /CS goes high, WEL is cleared.
READ	0000_0011b	2-byte*	Memory Data out	Reads data from F-RAM array
WRDI	0000_0100b	-	-	Clears WEL
RDSR	0000_0101b	-	St Reg data out	Read WPEN, BP(1:0), WEL bits
WRSR	0000_0001b	-	St Reg data in	Write WPEN and BP(1:0) bits
SLEEP	1011_1001b	-	-	Enter Sleep mode

\* Note that some devices use 1-byte and upper address bit(s) in the op-code and the 1Mb & higher densities uses a 3-byte address. See Addressing an SPI F-RAM for details.

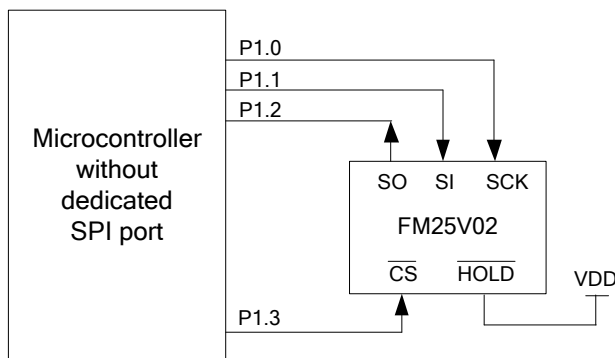
**SYSTEM HOOKUP**

Multiple devices may be used as long as the controller has extra pins to drive a chip select to each F-RAM device.



**Figure 3. System Configuration for Two F-RAM Devices**

For a microcontroller that has no dedicated SPI bus, a general purpose port may be used. Figure 4 shows this configuration. Bit-banging code drives this interface.



**Figure 4. System Configuration for µC using GPIO – Single F-RAM**

**STAND-ALONE SPI F-RAM PRODUCTS**

The following table summarizes the basic characteristics of the stand-alone SPI F-RAM products. The parts designated with a “B” are specified to operate over a 2.7 to 3.6V range and the V-family devices operate from 2.0V to 3.6V. The 5V “B” devices operate over a 4.5 to 5.5V. The new FM25W256 operates over a wide operating range, 2.7V to 5.5V.

**Table 2. SPI F-RAM Product Lineup**

	3V								5V			
	FM25L04B	FM25L16B	FM25CL64B	FM25V(N)01	FM25V(N)02	FM25V(N)05	FM25V(N)10	FM25H20	FM25040B	FM25C160B	FM25640B	FM25W256
Density	4Kb	16Kb	64Kb	128Kb	256Kb	512Kb	1Mb	2Mb	4Kb	16Kb	64Kb	256Kb
Organized Internally	512x8	2Kx8	8Kx8	16Kx8	32Kx8	64Kx8	128Kx8	256Kx8	512x8	2Kx8	8Kx8	32Kx8
Number of address bits	9	11	13	14	15	16	17	18	9	11	13	15
Number of address bytes	1	2	2	2	2	2	3	3	1	2	2	2
Operating Voltage	2.7-3.6V	2.7-3.6V	2.7-3.6V	2.0-3.6V	2.0-3.6V	2.0-3.6V	2.0-3.6V	2.7-3.6V	4.5-5.5V	4.5-5.5V	4.5-5.5V	2.7-5.5V
Max. Clock Freq.	20MHz	20MHz	20MHz	40MHz	40MHz	40MHz	40MHz	40MHz	20MHz	20MHz	20MHz	20MHz
Supported Clock Modes	0, 3	0, 3	0, 3	0, 3	0, 3	0, 3	0, 3	0, 3	0, 3	0, 3	0, 3	0, 3
Sleep Mode				✓	✓	✓	✓	✓				
Unique S/N				✓	✓	✓	✓					
Device ID				✓	✓	✓	✓					
Package	SOIC8 DFN8 (4x4.5)	SOIC8 DFN8 (4x4.5)	SOIC8 DFN8 (4x4.5)	SOIC8	SOIC8 DFN8 (4x4.5)	SOIC8	SOIC8	EIAJ DFN8† (5x6)	SOIC8	SOIC8	SOIC8	SOIC8

† 5 x 6 mm DFN8 conforms to SOIC8 footprint

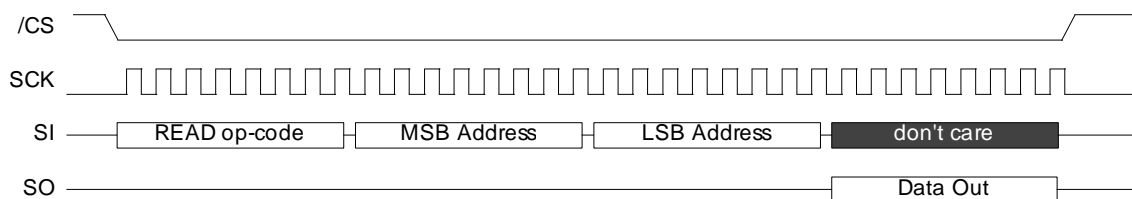
**READ/WRITE TRANSACTIONS**

The SPI interface is synchronous to a clock that is driven by the controller. All SPI devices that Ramtron builds will register data input on the rising edge of SCK and drive data back to the controller on the falling edge of SCK. To comply with this timing, controllers generally drive signals to the memory on the falling edge of SCK so that the signals have time to propagate and satisfy the setup timing specifications of the memory device.

### Memory Reads

Format: READ op-code, MSB Address, LSB Address, Data-out, (Data-out, Data-out, ...)

A read cycle is straightforward. The controller issues a READ op-code and address, then data comes out on the SO pin, e.g. data-out(0), data-out(1), data-out(2), etc. The /CS pin must remain low throughout the cycle. Once /CS is deasserted high, data output stops and SO goes to a hi-Z state. The clocked-in address is the starting address of the first data byte. Subsequent data bytes may be accessed simply by keeping /CS low while clocking-out data byte after data byte, each byte being read from an address incremented by the SPI F-RAM device. Figure 5 shows a two-byte address which is used for 16Kb through 512Kb densities.



**Figure 5. Read SPI Timing**

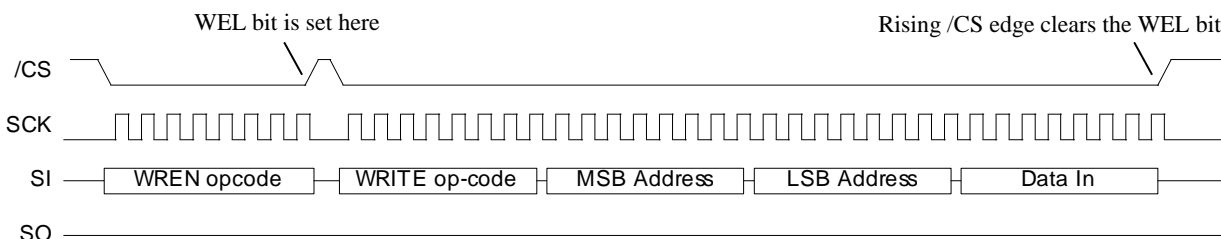
### Memory Writes:

Format: WREN op-code, WRITE op-code, MSB Address, LSB Address, Data-in, (Data-in, Data-in, ...)

A write cycle requires the controller to issue two op-codes, WREN and WRITE, in the following sequence. Each op-code must be bounded by /CS low. WREN op-code, WRITE op-code, address, data-in(0), data-in(1), data-in(2), etc. The clocked-in address is the starting address of the first data byte. Subsequent data bytes may be written by keeping /CS low while clocking-in data byte after data byte, each byte being written to an address incremented by the SPI F-RAM device. Each data byte is written to the F-RAM array on the 8<sup>th</sup> clock edge of that byte. There are no page buffers or write delays.

Note that the WEL bit in the Status Register is internally set and cleared by the SPI F-RAM device. It is set after clocking-in the WREN op-code and is cleared on the rising edge of /CS at the end of a write operation. Reading the Status Register (RDSR op-code) between the WREN and WRITE op-codes will not clear the WEL bit. Some users read the Status Register immediately following the WREN to check that the WEL bit is set. However, reading the WEL bit is not necessary to complete the write operation.

A complete single-byte write transaction is shown in Figure 6. This shows a two-byte address which is used for 16Kb through 512Kb densities.

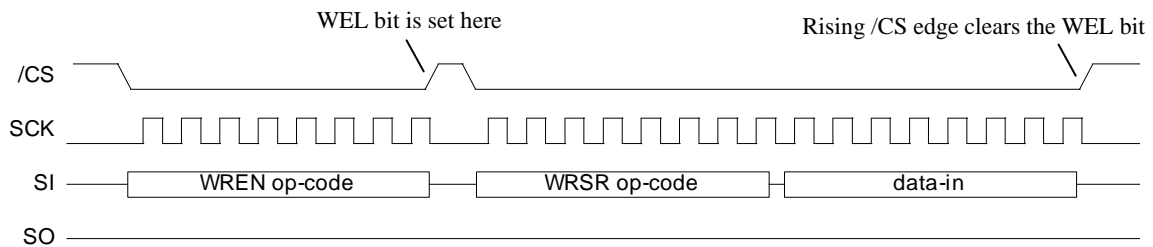


**Figure 6. Write SPI Timing**

Status Register Write:

*Format: WREN op-code, WRSR op-code, Data-in*

Writing the Status Register allows the user to 1) write-protect memory blocks and 2) enable the /WP pin. There are two block protect bits BP1 and BP0. They provide upper quarter, upper half, or entire array protection against writes. Once the BP(1:0) bits are written, the states are retained even after chip power is removed. WPEN enables or disables the external /WP pin. Software may be used to override the /WP pin from system tampering. WEL is a read-only bit that simply tells the user if the Write Enable Latch has been set, which allows writes to either the Status Register or the memory.

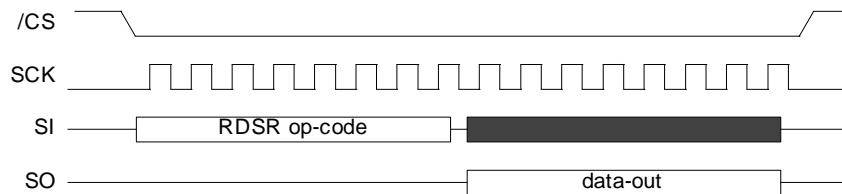


**Figure 7. Write Status Register Timing**

Status Register Read:

*Format: RDSR op-code, Data-out*

Reading the Status Register allows the user to view the state of the WPEN bit, the BP(1:0) write-protect bits, and the WEL bit.



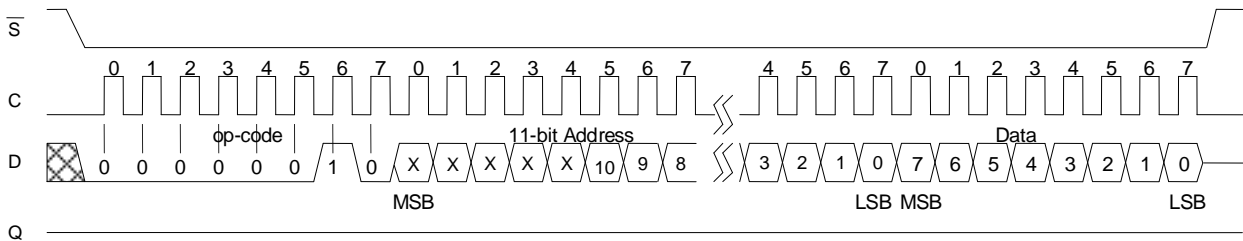
**Figure 8. Read Status Register Timing**



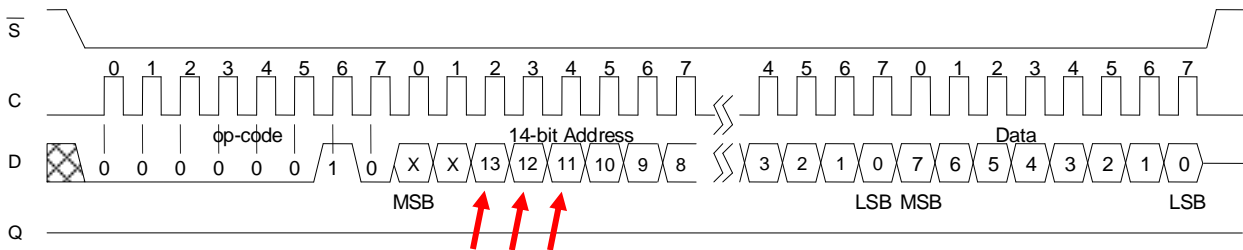
## PLACING A HIGHER DENSITY DEVICE IN A LOW DENSITY SOCKET

In the event that a particular device is not available and the board/system is designed for a lower density F-RAM device, it is possible to substitute with a higher density device. For example, a system that is designed to use a 16Kb device can also use a 64Kb, 128Kb, 256Kb, or 512Kb device. You can see from the previous figure that these devices use a common 2-byte address for the read and write operations. The devices within these densities are identical in terms of pinout, package (SOIC), and read/write functionality, assuming one complies with the specified operating voltage and timing requirements. There are two potential issues that may be a problem. If the system uses (1) the address wrap feature in the device, or (2) uses the block protect feature. In both cases, remember that a 16Kb device wraps at 0x800, a 64Kb device wraps at 0x2000, a 128Kb device address wraps at 0x4000, and so on. The block-protect boundaries are spaced at twice the address, when comparing device densities that are 2x from each other.

For example, the following two figures show the differences in the serial address streams between 16Kb and 128Kb devices.



**Figure 10. FM25L16B Write Cycle (WREN not shown)**



**Figure 11. FM25V01 Write Cycle (WREN not shown)**

When comparing a 16Kb and 128Kb device address requirements in Figure 9, one can see that the 3 additional address bit locations (A13, A12, A11) are used on a 128Kb device. As long as the controller drives these 3 address bits consistently for both reads and writes, a higher density device will work in a system designed for a lower density part.

A 1Mb density (or higher) device uses a 3-byte address and are not candidates as replacement parts in systems that are designed for lower densities. For example, a system that issues a 2-byte address will not work properly if a 3-byte address memory is used. See Table 2 (page 4) for information on the number of address bytes by density.

## CLOCKING MODES

The FM25xxx families support two of the four clocking modes: Mode 0 and Mode 3. Some of Ramtron’s early SPI parts support only Mode 0. Note that independent of the mode, all Ramtron parts clock data into the device on the rising SCK edge and clock data out on the falling edge of SCK. The difference between Modes 0 and 3 is simply whether SCK starts low or high when /CS is asserted low.

	Mode 0	Mode 1	Mode 2	Mode 3
SCK Starts ...	Low	Low	High	High
SI Data-In Latched on ...	↑	↓	↓	↑
SO Data-Out Driven from ...	↓	↑	↑	↓

Ramtron F-RAM devices support Modes 0 and 3.

## HALF-DUPLEX OPERATION

To reduce pincount on an SPI interface, the data lines can be tied together to create a common data I/O line. This 3-wire interface is SPI’s minimum pincount configuration. The controller must now ensure that the SIO line is hi-Z during read cycles. A secondary issue is that since the data bus is now half-duplex, this potentially reduces the data bandwidth.

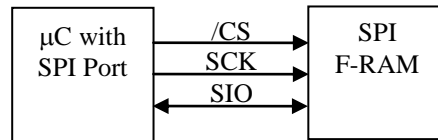


Figure 12. True 3-Wire SPI Interface

In this configuration, the data line is bi-directional and the controller must be able to hi-Z the data line when the memory device is driving read data back to the host. Bus contention will otherwise occur. Regardless of the timing mode (0 or 3), all SPI F-RAMs latch data-in on the rising edge of SCK and drive data-out on the falling edge of SCK.

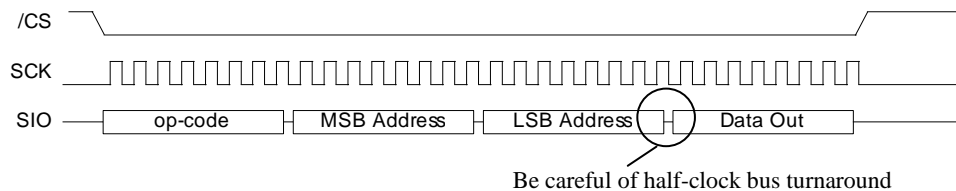


Figure 13. Common Data I/O Line and Bus Turnaround

## WRITE PROTECTION

SPI F-RAM devices may be write-protected using the /WP hardware pin or programming Status Register bits. The Status Register itself may be protected in addition to the F-RAM memory array. The Status Register contains nonvolatile Block Protect bits BP(1:0) to disable writes to portions of the memory array. The BP bits are colored yellow to indicate they are nonvolatile. There is also a nonvolatile bit called WPEN which enables the /WP hardware pin. Bit location 0 is reserved as a /RDY bit for EEPROM and serial flash. This bit is used in these devices so that the user can determine whether or not the memory is ready for another command by reading the Status Register. The /RDY bit is internally hardwired low in all SPI F-RAM devices since the chip is always ready after a write cycle.

A Write Protection table is provided in every SPI F-RAM datasheet. It covers all the cases for write protecting the Status Register and the F-RAM array. Here's a summary of the table:

- F-RAM array is protected by BP bit(s) high, even if WPEN=0 and /WP pin=1
- F-RAM array is not protected when BP bit(s) low, even if WPEN=1 and /WP pin=0
- Status Register is protected only when WPEN=1 and /WP pin=0
- Since the FM25040B and FM25L04B do not have a WPEN bit, all writes (memory array and Status Register) are blocked when /WP pin=0

Status Register							
7	6	5	4	3	2	1	0
WPEN	0	0	0	BP1	BP0	WEL	0

Figure 14. Status Register and Block Protect Settings

## POWER CYCLING

An F-RAM device is a high speed non-volatile memory and if it is active during a power ramp up or ramp down, there is the possibility of overwriting array data. Care must be taken in controlling signals during power cycle events. It is important to understand the device might inadvertently write data at mid-level power supply levels when chip select is active (low), for example.

SPI F-RAM devices have no power management circuits other than a simple internal power-on reset circuit. It is the user's responsibility to ensure that  $V_{DD}$  is within datasheet tolerances to prevent incorrect operation. It is recommended that the  $V_{DD}$  power supply voltage ramp up and ramp down in a well-controlled manner. Switch-mode power supplies are notorious for uncontrolled outputs as they power-up or power-down. It is also recommended that the device is powered down with chip select inactive (high). The system designer should be aware of chip-enable and  $V_{DD}$  states during power cycles. Please refer to application note, "[AN302 – SPI Reads/Writes and Data Protection During Power Cycles](#)". It describes the internal memory operations and offers some system design suggestions to avoid data corruption. It can be found on the Ramtron web site.

## Pseudo Code Examples (1-byte address, 4Kb devices)

```
#define WREN 0x06
#define WRITE 0x02 // Access lower half of memory
#define WRITE 0x0A // Access upper half of memory
#define READ 0x03 // Access lower half of memory
#define READ 0x0B // Access upper half of memory
#define RDSR 0x05
#define WRSR 0x01
#define WRDI 0x04
```

*In every case below, the parentheses designate the /CS pin going low (“ and high “).*

```
/****** Memory Write (single byte to location 0130h) *****/
WREN (0x06) // Sets WEL bit. WREN must precede WRITE opcode.
WRITE (0x0A, // 0x02 is WRITE opcode and A8 bit set
      0x30, // starting address
      0x55) // 0x55 is data written to location 0130h

/****** Memory Write (multiple bytes to starting location 01FCh) *****/
WREN (0x06) // Sets WEL bit. WREN must precede WRITE opcode.
WRITE (0x0A, // 0x02 is WRITE opcode and A8 bit set
      0xFC, // starting address
      0x55, // 0x55 is data written to location 01FCh
      0xAA, // 0xAA is data written to location 01FDh
      0x55, // 0x55 is data written to location 01FEh
      0xAA) // 0xAA is data written to location 01FFh

/****** Memory Read (single byte from location 01D3h) *****/
READ (0x0B, // 0x03 is READ opcode
     0xD3, // starting address
     0xAA) // 0xAA is data read from location 01D3h

/****** Memory Read (multiple bytes from starting location 01FCh) *****/
READ (0x0B, // 0x03 is READ opcode
     0xFC, // starting address
     0x55, // 0x55 is data read from location 01FCh
     0xAA, // 0xAA is data read from location 01FDh
     0x55, // 0x55 is data read from location 01FEh
     0xAA) // 0xAA is data read from location 01FFh

/****** Write Status Register (write protect upper half of memory) *****/
WREN (0x06) // Sets WEL bit. WREN must precede WRSR opcode.
WRSR (0x01, // 0x01 is WRSR opcode
     0xF8) // 0xF8 sets the BP1 bit which protects the upper
           // half of the memory array. The upper nibble
           // set to "F" attempts to write the upper bits
           // to 1.

/****** Read Status Register *****/
RDSR (0x05, // 0x05 is RDSR opcode
     0x08) // 0x08 tells us that the BP1 bit is set and that
           // the upper half of the memory array is protected.
           // The upper nibble returns "0" since they are
           // hardwired low.
```

NOTE: Text in **BLUE** indicates data being sent by the controller. Text in **RED** indicates data being received by the controller.

## Pseudo Code Examples (2-byte address, 16Kb through 512Kb devices)

```
#define WREN 0x06
#define WRITE 0x02
#define READ 0x03
#define RDSR 0x05
#define WRSR 0x01
#define WRDI 0x04
```

*In every case below, the parentheses designate the /CS pin going low (“ and high “).*

```

/***** Memory Write (single byte to location 0F30h) *****/
WREN (0x06)           // Sets WEL bit. WREN must precede WRITE opcode.
WRITE (0x02,         // 0x02 is WRITE opcode
      0x0F,         // starting address MSB
      0x30,         // starting address LSB
      0x55)         // 0x55 is data written to location 0F30h

/***** Memory Write (multiple bytes to starting location 07FC) *****/
WREN (0x06)           // Sets WEL bit. WREN must precede WRITE opcode.
WRITE (0x02,         // 0x02 is WRITE opcode
      0x07,         // starting address MSB
      0xFC,         // starting address LSB
      0x55,         // 0x55 is data written to location 07FCh
      0xAA,         // 0xAA is data written to location 07FDh
      0x55,         // 0x55 is data written to location 07FEh
      0xAA)         // 0xAA is data written to location 07FFh

/***** Memory Read (single byte from location 0F31h) *****/
READ (0x03,          // 0x03 is READ opcode
     0x0F,          // starting address MSB
     0x31,          // starting address LSB
     0xAA)          // 0xAA is data read from location 0F31h

/***** Memory Read (multiple bytes from starting location 07FCh) *****/
READ (0x03,          // 0x03 is READ opcode
     0x07,          // starting address MSB
     0xFC,          // starting address LSB
     0x55,          // 0x55 is data read from location 07FCh
     0xAA,          // 0xAA is data read from location 07FDh
     0x55,          // 0x55 is data read from location 07FEh
     0xAA)          // 0xAA is data read from location 07FFh

/***** Write Status Register (write protect upper half of memory) *****/
WREN (0x06)           // Sets WEL bit. WREN must precede WRSR opcode.
WRSR (0x01,         // 0x01 is WRSR opcode
     0x08)         // 0x08 sets the BP1 bit which protects the upper
                    // half of the memory array.

/***** Read Status Register *****/
RDSR (0x05,         // 0x05 is RDSR opcode
     0x88)         // 0x88 tells us that the BP1 bit is set and that
                    // the upper half of the memory array is protected.
                    // The WPEN bit is also set which works with
                    // the /WP pin to protect the Status Register.

```

NOTE: Text in **BLUE** indicates data being sent by the controller. Text in **RED** indicates data being received by the controller.

## Pseudo Code Examples (3-byte address for FM25V10 and FM25H20)

```
#define WREN 0x06
#define WRITE 0x02
#define READ 0x03
#define RDSR 0x05
#define WRSR 0x01
#define WRDI 0x04
```

*In every case below, the parentheses designate the /CS pin going low (“ and high “).*

```

/***** Memory Write (single byte to location 1BF30h) *****/
WREN (0x06)           // Sets WEL bit. WREN must precede WRITE opcode.
WRITE (0x02,         // 0x02 is WRITE opcode
      0x01,         // starting address MSB
      0xBF,         // starting address 2nd byte
      0x30,         // starting address LSB
      0x55)         // 0x55 is data written to location 1BF30h

/***** Memory Write (multiple bytes to starting location 1B7FC) *****/
WREN (0x06)           // Sets WEL bit. WREN must precede WRITE opcode.
WRITE (0x02,         // 0x02 is WRITE opcode
      0x01,         // starting address MSB
      0xB7,         // starting address 2nd byte
      0xFC,         // starting address LSB
      0x55,         // 0x55 is data written to location 1B7FCh
      0xAA,         // 0xAA is data written to location 1B7FDh
      0x55,         // 0x55 is data written to location 1B7FEh
      0xAA)         // 0xAA is data written to location 1B7FFh

/***** Memory Read (single byte from location 1BF31h) *****/
READ (0x03,          // 0x03 is READ opcode
      0x01,          // starting address MSB
      0xBF,          // starting address 2nd byte
      0x31,          // starting address LSB
      0xAA)         // 0xAA is data read from location 1BF31h

/***** Memory Read (multiple bytes from starting location 1B7FCh) *****/
READ (0x03,          // 0x03 is READ opcode
      0x01,          // starting address MSB
      0xB7,          // starting address 2nd byte
      0xFC,          // starting address LSB
      0x55,          // 0x55 is data read from location 1B7FCh
      0xAA,          // 0xAA is data read from location 1B7FDh
      0x55,          // 0x55 is data read from location 1B7FEh
      0xAA)         // 0xAA is data read from location 1B7FFh

/***** Write Status Register (write protect upper half of memory) *****/
WREN (0x06)           // Sets WEL bit. WREN must precede WRSR opcode.
WRSR (0x01,         // 0x01 is WRSR opcode
      0x08)         // 0x08 sets the BP1 bit which protects the upper
                    // half of the memory array.

/***** Read Status Register *****/
RDSR (0x05,          // 0x05 is RDSR opcode
      0x88)         // 0x88 tells us that the BP1 bit is set and that
                    // the upper half of the memory array is protected.
                    // The WPEN bit is also set which works with
                    // the /WP pin to protect the Status Register.

```

NOTE: Text in **BLUE** indicates data being sent by the controller. Text in **RED** indicates data being received by the controller.

**Revision History**

<b>Revision</b>	<b>Date</b>	<b>Summary</b>
1.0	1/15/2007	Initial Release
1.1	2/25/2008	Updated part numbers (i.e. FM25L256 to FM25L256B) and attributes. Fixed pseudo-code page.
1.2	5/16/2008	Added FM25H20 and attributes.
1.3	9/30/2009	Added FM25V10/V05/V02 and attributes. Added 3-byte addressing pseudo-code examples.
1.4	2/18/2011	Updated part numbers. Added section on substituting a higher density device in a system that uses a lower density. Added 1-byte address example code.