Please note that Cypress is an Infineon Technologies Company.
The document following this cover page is marked as “Cypress” document as this is the company that originally developed the product. Please note that Infineon will continue to offer the product to new and existing customers as part of the Infineon product portfolio.

Continuity of document content
The fact that Infineon offers the following product as part of the Infineon product portfolio does not lead to any changes to this document. Future revisions will occur when appropriate, and any changes will be set out on the document history page.

Continuity of ordering part numbers
Infineon continues to support existing part numbers. Please continue to use the ordering part numbers listed in the datasheet for ordering.

AN408

A Design Guide to SPI F-RAM™ Processor Companion - FM33256B

Author: Harsha Medu
Associated Part Family: FM33256B
Associated Code Examples: CE204087
Related Application Notes: click here

AN408 provides an overview and design guidelines for the SPI F-RAM real-time clock (RTC) Processor Companion part - FM33256B. This document also includes a typical application and an example code. For information on the basic SPI operation, refer to the application note AN304 - SPI Guide for F-RAM.

Contents

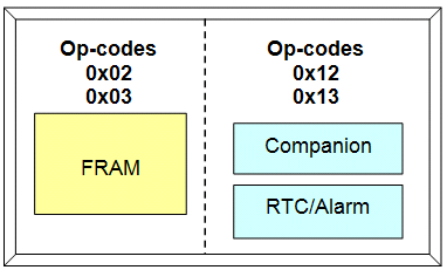
1 Introduction

The FM33256B is a 256-Kbit serial (SPI) F-RAM memory product, which offers an integrated processor companion and an RTC. The processor companion section includes a power-on system reset, low-voltage detect, a watchdog timer, an early power-fail warning, an event counter, automatic switchover to backup power, and a lockable 64-bit serial number. It employs an industry-standard SPI, which is used to access the memory, the processor companion, and the RTC. The FM33256B operates over a 2.7 V-to-3.6 V power supply range with the maximum SPI frequency of 16 MHz. It is available in an SOIC-14 package.

2 Two Logical Devices in One

Even though FM33256B is a single device, it is internally organized as two logical devices as shown in Figure 1.

Figure 1. Two Logical Devices with Unique Opcodes



The memory is one logical device, and the companion/RTC is the other logical device. This helps to integrate both the functionalities without affecting each other. Each has its own address space and is accessed via the SPI. The F-RAM is accessed through the standard memory opcodes (0x02 for Write and 0x03 for Read). The processor companion/RTC is accessed through special opcodes (0x12 for Write and 0x13 for Read).

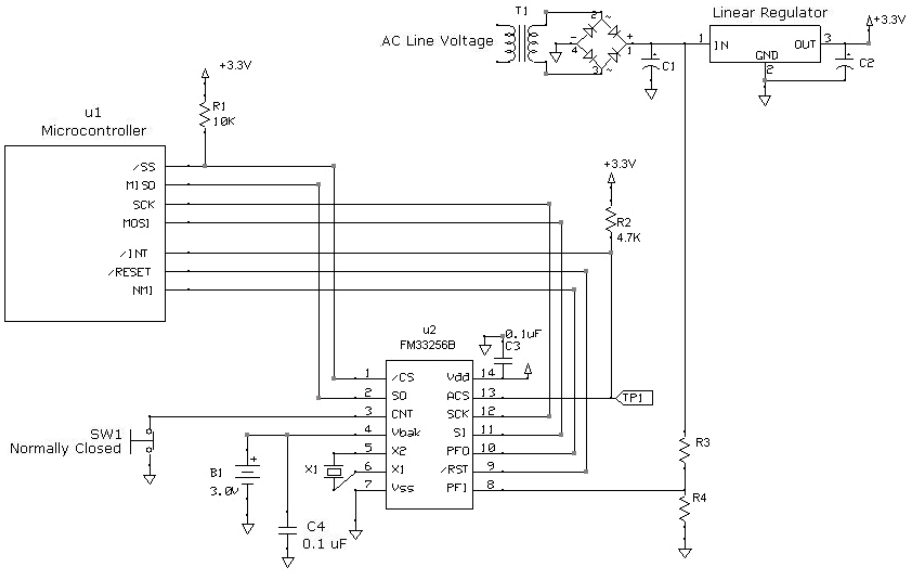
Note: For the complete list of opcodes, refer to Table 2.

3 Typical Application

A typical application of FM33256B is shown in Figure 2. It is shown as an example with external components, their typical values, and connections to other system devices. This application is a line-operated (AC powered) system with a microcontroller (the FM33256B device), and other passive components. The example microcontroller has a dedicated SPI port. All microcontrollers may not have this port; in such cases, the SPI protocol may be implemented in firmware and bit-banged through GPIO pins.

Note: Bit banging is a technique used for serial communications. It uses firmware instead of a dedicated hardware. Firmware directly sets and samples the state of pins on the microcontroller, and is responsible for all parameters of the signal like timing, levels, synchronization and so on.

Figure 2. FM33256B Reference Schematic



4 Processor Companion Features

The FM33256B device integrates all the necessary processor supervisor features that a designer may need.

The companion features include:

- System Power-On Reset (POR) with $\overline{\text{RST}}$ pin
  - Low -Voltage Detect
  - Watchdog
  - Manual reset
- Early power-fail warning
- Event counters
- Lockable serial number

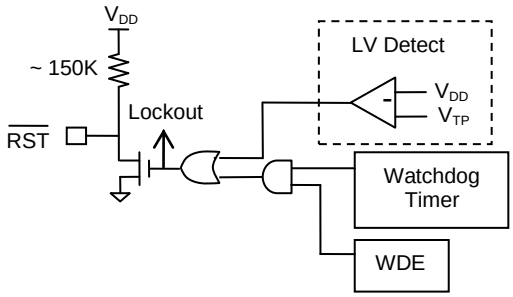
The following sections describe each of these in detail.

4.1 System Power-On Reset with $\overline{RST}$ pin

The FM33256B product provides a processor-reset signal when the system powers up and whenever a system fault or manual override (manual hardware reset) occurs. The $\overline{RST}$ pin is primarily used to drive a reset to the processor, but can be used as an input to provide a hardware reset to the system. The $\overline{RST}$ pin does not reset or clear any FM33256B internal registers.

There are two trigger sources that can drive reset LOW: a Low Voltage (LV) Detect circuit and the Watchdog Timer as shown in Figure 3.

Figure 3. Reset Trigger Sources



At power-up, the $\overline{RST}$ pin is driven LOW as V_{DD} rises toward its nominal operating value. The point at which the $\overline{RST}$ pin is released is determined by V_{TP} (Voltage Trip Point), an internal trip voltage that is always compared to V_{DD}. The internal pull-up resistor (approximately 150 kΩ) on the $\overline{RST}$ pin eliminates the need for an external resistor. When tripped, the reset circuit times out after approximately 65 ms (30 ms minimum, 100 ms maximum). You may set V_{TP} (2 bits, VTP1, VTP0 in register 18h) to one of four values: 2.6 V, 2.75 V, 2.9 V, or 3.0 V.

The other source of trigger for the $\overline{RST}$ pin is the watchdog timer, a free-running, user-programmable timer that can be set to time out as soon as 60 ms or as long as 1.8 seconds. The timer settings are stored in nonvolatile registers (registers 0Bh and 0Ch), so there is no need to reinitialize these values or the trip voltage setting again. The watchdog timer is controlled by the enable bit WDE (register 0Ch, bit 7). The Watchdog Timer works as a window timer which allows for a delayed start.

There are two ways for the watchdog timer to trigger a reset. First, the Early Watchdog Timer Fault (EWDF in register 09h, bit 7) forces a reset condition when a restart occurs too early (before the programmed start time (register 0Bh) for the Watchdog Timer window is reached). Second, the Late Watchdog Timer Fault (LWDF) forces a reset condition when the timer restart occurs too late (the programmed end time (register 0Ch) for the Watchdog Timer window is exceeded). The timer may be restarted at any time within the timeout window by writing 1010b to the Watchdog Restart register (register 0Ah).

When a reset condition occurs, the device will set one of the three flags to indicate the source of the reset in the register 09h: EWDF, LWDF, and POR. These bits are battery-backed during power-down. After power is restored, the system host can read these bits to determine the cause of the reset. The host must clear these bits after reading them.

A manual hardware reset may be provided in the system by simply connecting a momentary contact switch to the $\overline{RST}$ pin. The $\overline{RST}$ pin detects an external LOW input condition and responds by driving the $\overline{RST}$ signal LOW for 100 ms (maximum). This effectively filters and debounces a reset switch. Note that in all reset cases, the chip internally drives a Lockout signal that serves to block any SPI traffic and abort any pending write accesses to the F-RAM.

4.2 Early Power-Fail Warning

The F-RAM integrated Processor Companion features a general-purpose comparator that can be used to generate an early power-fail interrupt (PFI). This warning signal can drive a microcontroller interrupt input and must occur before V_{DD} drops too low to save all critical data to the nonvolatile RAM. The processor companion's early power-fail interrupt can also be used as a warning to the system to stop conducting critical activity during a brownout condition. For more information, refer to the application note AN400 - Generating a Power-Fail Interrupt using the F-RAM Processor Companion.

4.3 Event Counter

The FM33256B device integrates a 16-bit event counter for tamper detection or other event-logging purposes. The counter has an input pin (CNT) that is edge-triggered and polarity that is user-defined. The event, or edge, must be CMOS-logic level. The counter control register is nonvolatile and counter values are either battery-backed or nonvolatile based on the Nonvolatile/Volatile Counter (NVC, in register 0Dh, bit 7) user selection.

The event counter can be operated in two modes.

- Continuous mode (POLL = 0, register 0Dh, bit 1)
- Polled mode (POLL = 1, register 0Dh, bit 1)

In continuous mode, the CNT pin is continuously monitored for any event and the event counter is incremented when an event is detected. The polarity of the event can be either a rising edge (Counter Polarity bit, CP = 1, register 0Dh, bit 0) or a falling edge (CP = 0, register 0Dh, bit 0). The events which occur as fast as 1 KHz can be detected in this mode. Depending on the application, you can set the event counter to work on either V_{DD} or V_{BAK}. If events occur when power is down, event counter should work on V_{BAK} (NVC = 0 in register 0Dh, bit 7). Otherwise, it can be configured to work on V_{DD} (NVC = 1 in register 0Dh, bit 7) which will result in longer battery life.

Note: In continuous mode, a pull-up resistor on CNT pin should be connected appropriately to either the backup supply V_{BAK} or V_{DD}.

Continuous mode is not suitable for detecting tamper events which are generally one-time or low-frequency events and can occur when power is down. In such cases, polled mode can be used. Polled mode works only with battery-backed (NVC = 0, register 0Dh, bit 7) and rising- edge detection (CP = 1, register 0Dh, bit 0) configuration. In polled mode, the CNT pin is polled once every 125 ms resulting in a reduced power consumption and longer battery life. The value in the event counter may not represent the actual number of tamper events, but a positive value indicates a tamper.

Note: In polled mode, the CNT pin has an internal pull-up resistor and therefore no external pull-up is required.

A typical application in Figure 2 shows a normally-closed switch connected between the CNT pin and ground (or a system case or chassis). The CNT pin is set to polled mode, which occasionally samples the pin in order to minimize the battery drain. There are no external resistors required for this case. The event counter is configured to detect a rising edge (CP = 1, register 0Dh, bit 0) on the CNT pin. When the switch is open, a tamper event is registered and the counter increments.

4.4 Serial Number

The F-RAM processor companion provides a 64-bit lockable serial number (register 10h to register 17h in Table 1). The serial number provides a unique way of identifying each product. The serial number has unlimited writes until the lock-bit is set by the user.

5 Real-Time Clock

A real-time clock (RTC) counts time in steps of seconds and can report time, day of the week, calendar day, week, and year. It can be used to record time when certain system events occur. The RTC consists of an oscillator and counters that derive the time/calendar information, and several registers that hold the time and RTC control settings. The RTC will run continuously even if the main power fails as long as a 3-V backup power source is connected to the V_{BAK} input pin. A backup source can be either a 3-V battery or a super capacitor.

The default factory settings disable the RTC oscillator. To start and configure the RTC, the $\overline{\text{OSCEN}}$ bit (register 00h, bit 7) must first be set to ‘0’. Then, the clock and calendar registers must be written to reflect the current time, day, and date. The RTC register map is shown in Table 1.

Table 1. RTC Register Map

Battery-backed = Nonvolatile = BB/NV User Programmable =

Address	Data								Function	Range
	D7	D6	D5	D4	D3	D2	D1	D0		
1Dh	$\overline{M}$	0	0	Alarm 10 months	Alarm months				Alarm Month	01-12
1Ch	$\overline{M}$	0	Alarm 10 date		Alarm date				Alarm Date	01-31
1Bh	$\overline{M}$	0	Alarm 10 hours		Alarm hours				Alarm Hours	00-23
1Ah	$\overline{M}$	Alarm 10 minutes			Alarm minutes				Alarm Minutes	00-59
19h	$\overline{M}$	Alarm 10 seconds			Alarm seconds				Alarm Seconds	00-59
18h	SNL	AL/SW	F1	F0	VBC	FC	VTP1	VTP0	Companion Control	
17h	Serial Number Byte 7								Serial Number 7	FFh
16h	Serial Number Byte 6								Serial Number 6	FFh
15h	Serial Number Byte 5								Serial Number 5	FFh
14h	Serial Number Byte 4								Serial Number 4	FFh
13h	Serial Number Byte 3								Serial Number 3	FFh
12h	Serial Number Byte 2								Serial Number 2	FFh
11h	Serial Number Byte 1								Serial Number 1	FFh
10h	Serial Number Byte 0								Serial Number 0	FFh
0Fh	Event Counter Byte 1								Event Counter 1	FFh
0Eh	Event Counter Byte 0								Event Counter 0	FFh
0Dh	NVC	-	-	-	RC	WC	POLL	CP	Event Counter Control	
0Ch	WDE	-	-	WDET4	WDET3	WDET2	WDET1	WDET0	Watchdog Control	
0Bh	-	-	-	WDST4	WDST3	WDST2	WDST1	WDST0	Watchdog Control	
0Ah	-	-	-	-	WR3	WR2	WR1	WR0	Watchdog Restart	
09h	EWDF	LWDF	POR	LB	-	-	-	-	Watchdog Flags	
08h	10 years				years				Years	00-99
07h	0	0	0	10 months	months				Month	01-12
06h	0	0	10 date		date				Date	01-31
05h	0	0	0	0	0	day			Day	01-07
04h	0	0	10 hours		hours				Hours	00-23
03h	0	10 minutes			minutes				Minutes	00-59
02h	0	10 seconds			seconds				Seconds	00-59
01h	-	-	CALS	CAL4	CAL3	CAL2	CAL1	CAL0	CAL/Control	
00h	$\overline{OSCEN}$	AF	CF	AEN	reserved	CAL	W	R	RTC/Alarm Control	

The 32.768-kHz oscillator is divided down through a series of counters. The first counter divides it by 32,768 to derive the 1-Hz signal for the seconds counter. The next counter uses the seconds counter to drive a signal once per minute to the minutes counter. Subsequent counters continue to divide down until there is one pulse per hour, month, and year driving their respective counters. The counters hold the time and date information that is memory-mapped (locations 02h through 08h) into the RTC address space. Because the FM33256B uses different opcodes for the companion register space, they do not appear as F-RAM locations. You may read and write the time and date by reading and writing these locations without affecting the F-RAM locations.

In the FM33256B, the RTC data may be read without interrupting the RTC operation. When the RTC is read by writing the ‘R’ bit in the control register (register 00h, bit 0) with a ’1’, a snapshot is taken of the current time-day-date and stored in the registers where it can be read by the host. The snapshot ensures that the time doesn’t change between successive reads from the host. However, internally, the RTC will be running normally. Similarly, during RTC writes, the registers hold the data written by the host and wait to transfer it to the RTC counters after all the time-day-date information has been written and the ‘W’ bit (register 00h, bit 1) is cleared. If W = 0, writes to the RTC registers are ignored. For more information on RTC read and write, refer to Setup Example

5.1 Backup Power

The RTC, event counter, and various control settings are maintained by using a backup power source on the V_{BAK} pin even if the primary power source is removed. The V_{BAK} source may be a battery or a large-value super capacitor. For more information, refer to AN404 - F-RAM RTC Backup Supply (V_{BAK} pin) and UL Compliance.

When using a super capacitor as the backup source, the capacitor can discharge over a long period to a point where the V_{BAK} voltage drops out of spec and the RTC stops functioning, resulting in the loss of backup data. To avoid this situation, the charge on the capacitor needs to be restored. For different types of charging, refer to AN401 - Charging Methods for the F-RAM RTC Backup Capacitor.

5.2 RTC Calibration

RTC calibration is required primarily to compensate the frequency shift due to crystal tolerance, temperature effect, and load-capacitance mismatch. A 512-Hz signal brought out on the ACS (Alarm/Calibration/SquareWave) pin is used for calibration. For more information about calibration, refer to AN402 - F-RAM RTC Oscillator Design Guide.

5.2.1 Setup Example

The following example is a setup procedure that describes the correct sequence for power-up, initialization, and register settings:

1. Apply V_{DD} power.
2. Apply battery to the V_{BAK} pin. If a super capacitor is used, the trickle charger may be set (VBC = 1). Optionally set the FC bit. If a battery is used, make VBC = 0.
3. Set the V_{DD} voltage trip point bit, VTP if you prefer a setting higher than the default setting (lowest of the four settings; 2.6 V).
4. Enable the RTC oscillator (set $\overline{\text{OSCEN}}$ = 0).
5. Enter the RTC calibration mode (CAL = 1).
6. Determine the RTC clock error and sign by monitoring the 512-Hz output (ACS pin). Use a frequency counter.
7. Write the error correction sign to Calibration Sign bit (CALS, register 01h, bit 5) and value to CAL (4:0). Refer to the Calibration Adjustments table provided in the datasheet.
8. Exit the RTC calibration mode (CAL = 0).
9. Set the Write bit (W bit) to enable writing the time-day-date values to the RTC.
10. Write the time-day-date values to registers 02h – 08h.
11. Clear the W bit to start the RTC with the new values.
12. Resume normal operation.

To read the RTC, follow this simple three-step procedure:

1. Set the Read (R bit in register 00h, bit 0) which takes a snapshot of the RTC registers (assumes R bit is previously at logic 0).
2. Issue the Read Processor Companion command (RDPC = 0x13), starting at the address 02h, and read seven RTC bytes (02h – 08h).
3. Clear the R bit to prepare for the next RTC read.

6 Power Cycle Considerations

To protect the F-RAM from corrupting the data during power cycles, it is recommended that the MCU's $\overline{\text{SS}}$ (SPI Slave Select) control pin be held inactive as V_{DD} powers up and powers down. In many cases, this may be as simple as a pull-up resistor R1 on the MCU's output pin that drives the FM33256B $\overline{\text{CS}}$ (Chip Select) pin. As the system microcontroller powers up, its outputs will tristate before the power supply reaches sufficient voltage to turn various internal circuits on, thereby allowing the pull-up resistor to keep the signal at V_{DD}. Similarly, at power-down, the V_{DD} voltage reaches a point where it allows the outputs to "let go", again allowing the pull-up resistor to do its job. For more information, refer to the application note AN302 - F-RAM SPI Read & Write and Data Protection During Power Cycles.

Apart from $\overline{CS}$, you should also consider the V_{DD} power-up ramp rate, power-down ramp rate, and power-up to first access specifications. The power-up ramp rate should be slower than 50 µs/V and the power-down ramp rate should be slower than 100 µs/V. FM33256B keeps the $\overline{RST}$ pin LOW for t_{RPU} time at power-up ($V_{DD} > V_{TP}$). Therefore, you should wait for 100 ms (max) before the device can be accessed at power-up.

7 Summary

This application note summarizes the features of the FM33256B device. This document also provides a typical application, design guidelines, and example pseudo codes for the FM33256B device.

8 Related Application Notes

- AN407 - A Design Guide to I2C F-RAM Processor Companions – FM31278, FM31276, FM31L278, and FM31L276
- AN400 - Generating a Power-Fail Interrupt using the F-RAM Processor Companion
- AN401 - Charging Methods for the F-RAM RTC Backup Capacitor
- AN402 - F-RAM RTC Oscillator Design Guide
- AN404 - F-RAM RTC Backup Supply (V_{BAK} pin) and UL Compliance

9 PSoC 3 User Module

The PSoC 3-based user module project to access F-RAM processor companion is provided with this application note. See the user guide provided with the project for more information.

10 Pseudo Code Examples

A summary of the op-codes of FM33256B is given in Table 2.

Table 2. Summary Table of Op-Codes

Name	Op-Code	Address	Data	Action
WREN	0000_0110b	-	-	Sets WEL
WRITE	0000_0010b	2 bytes	Memory data in	Writes data to F-RAM array if WEL = 1. When $\overline{CS}$ goes HIGH, WEL is cleared.
READ	0000_0011b	2 bytes	Memory data out	Reads data from F-RAM array
WRDI	0000_0100b	-	-	Clears WEL
RDSR	0000_0101b	-	Status Register data out	Read WPEN, BP(1:0), WEL bits
WRSR	0000_0001b	-	Status Register data in	Write WPEN and BP(1:0) bits
RDPC	0001_0011b	1 byte	Register data out	Companion/RTC register read
WRPC	0001_0010b	1 byte	Register data in	Companion/RTC register write

```
#define nvRAM_WREN  0x06
#define nvRAM_RTC_WRITE_CMD 0x12
#define nvRAM_RTC_READ_CMD 0x13
```

10.1 Enable RTC Oscillator

```
       /************************** Enable RTC Oscillator
       *******************************/
       data[0] = 0x00;   // Data for clearing the OSCEN bit

       WRITE_RTC (0x00,      // Sets the address to Register 00h
                  data,  // Writes data 0x00 which clears the OSCEN bit
                  0x01); // Number of bytes to be written
```

10.2 Set RTC Time/Date

```
       /************************** Set RTC time/date ******************************/

       // Step #1 Set W bit which allows writes to RTC registers
       data[0] = 0x02;   // Data for setting the W bit

       WRITE_RTC (0x00,      // Sets the address pointer to Register 00h
                  data,  // Writes data 0x02 which sets the W bit
                  0x01); // Number of bytes to be written

       // Step #2 Write time/date to RTC Registers
       data[0] = 0x00;   // Seconds set to 00
       data[1] = 0x10;   // Minutes set to 10
       data[2] = 0x14;   // Hours set to 14 (2 PM)
       data[3] = 0x03;   // Day set to the third day of the week
       data[4] = 0x04;   // Date set to the fourth day in March
       data[5] = 0x03;   // Month set to March
       data[6] = 0x08;   // Year set to 2008

WRITE_RTC (0x02,           // Sets the address pointer to Register 02h
                  data,              // Writes time/date
            0x07);          // Number of bytes to be written

       // RTC does not start to run yet.

       // Step #3 Clear W bit to start RTC with the exact time
       data[0] = 0x00;            // data=0x00 clears the W bit

WRITE_RTC (0x00,           // Sets the address pointer to Register 00h
           data,           // Data=0x00 clears the W bit to start RTC with time
defined
                           // in Step #2. The 8th clock of data byte defines the
actual
                           // start of the RTC.
           0x01);          // Number of bytes to be written
```

10.3 Set VTP Voltage Detect Trip Point

```
      /********************* Set VTP Voltage Detect Trip Point
      *************************/
      // From the factory, VTP bits are cleared to 0. If user wants to set trip point
      to
      // the higher setting, then write VTP bits to 11b in Reg 18h control register.

      data[0] = 0x03;   // Data for setting both the VTP bits

      WRITE_RTC (0x18,      // Sets the address pointer to Register 0Bh
                 data,  // Writes data 0x03 which sets the VTP bits
                 0x01); // Number of bytes to be written
```

10.4 Read RTC Registers

```
      /************************* Read RTC Registers ******************************/

      // Step #1 Set R bit which takes snapshot of RTC registers
      data[0] = 0x01;   // Data for setting the R bit

      WRITE_RTC (0x00,      // Sets the address pointer to Register 00h
                 data,  // Writes data 0x01 which sets the R bit
                 0x01); // Number of bytes to be written

      // Step #2 Read RTC Registers
READ_RTC (0x02,            // Sets the address pointer to Register 02h
                data,               // Data buffer to read RTC registers
          0x07);           // Number of bytes to be read

      // data buffer contains the following
      // 0x59 - Seconds
      // 0x15 - Minute
      // 0x14 - Hour
      // 0x03 - Third day of the week
      // 0x04 - Fourth day in March
      // 0x03 - March
      // 0x08 - 2008

      // Step #3 Clear R bit
      data[0] = 0x00;            // data=0x00 clears the W bit

WRITE_RTC (0x00,           // Sets the address pointer to Register 00h
           data,           // Data=0x00 clears the R bit to allow RTC read next time
           0x01);          // Number of bytes to be written
```

10.5 Read Event Counters

```
       /************************* Read Event Counters
       ******************************/

       // Step #1 Set RC bit which takes snapshot of both counter registers
       data[0] = 0x09;   // Data for setting the RC bit and keeps CP=1

       WRITE_RTC (0x0D,      // Sets the address pointer to Register 0Dh
                  data,  // Writes data 0x09 which sets the R bit
                  0x01); // Number of bytes to be written

       // Step #2 Read Event Counter
READ_RTC (0x0E,            // Sets the address pointer to Register 0Dh
                 data,               // Data buffer to read RTC registers
          0x02);           // Number of bytes to be read

       // data buffer contains the following
       // 0x1A - CounterByte0 reads out LSB 0x1A (decimal 26)
       // 0x00 - CounterByte1 reads out MSB 0x00

       // Step #3 Clear RC bit
       data[0] = 0x01;            // data=0x01 clears the RC bit

WRITE_RTC (0x0D,           // Sets the address pointer to Register 0Ch
           data,           // Data=0x01 clears the RC bit and keeps C1P=1
           0x01);          // Number of bytes to be written
```

10.6 SPI Processor Companion Write

```
/*********PSoC3 Based Pseudo Code for SPI Processor Companion Write **********/
uint8 WRITE_RTC (uint8 addr, uint8 *data_write_ptr, uint8 total_data_count)
{
   uint8 i;

   // Clear the Transmit Buffer
   nvRAM_SPI_1_SPIM_ClearTxBuffer();

   // Send Write Enable WREN command
   // Make chip select LOW CS = 0
   nvRAM_SPI_1_CS_Reg_Write(0);
   // Send Write Enable Command WREN
   nvRAM_SPI_1_SPIM_WriteTxData(nvRAM_WREN);
   // Wait for the transfer to complete
   while((nvRAM_SPI_1_SPIM_ReadTxStatus() & nvRAM_SPI_1_SPIM_STS_SPI_DONE) !=
   nvRAM_SPI_1_SPIM_STS_SPI_DONE);
   // Make chip select High CS = 1
   nvRAM_SPI_1_CS_Reg_Write(1);

   // Delay
   CyDelay(1);

   // Clear the Transmit Buffer
   nvRAM_SPI_1_SPIM_ClearTxBuffer();

   // Make chip select LOW CS = 0
   nvRAM_SPI_1_CS_Reg_Write(0);
   // Send Processor Companion Write Command
   nvRAM_SPI_1_SPIM_WriteTxData(nvRAM_RTC_WRITE_CMD);
   // Send Processor Companion Register address
   nvRAM_SPI_1_SPIM_WriteTxData((uint8)(addr));
   // Wait for the transfer to complete
   while((nvRAM_SPI_1_SPIM_ReadTxStatus() & nvRAM_SPI_1_SPIM_STS_SPI_DONE) !=
   nvRAM_SPI_1_SPIM_STS_SPI_DONE);
   // Send Processor Companion Register data
   for(i = 0; i < total_data_count; i++ )
   {
      nvRAM_SPI_1_SPIM_WriteTxData((uint8)(data_write_ptr[i]));
      while((nvRAM_SPI_1_SPIM_ReadTxStatus() & nvRAM_SPI_1_SPIM_STS_SPI_DONE)
!=
      nvRAM_SPI_1_SPIM_STS_SPI_DONE);
   }
   // Make chip select HIGH CS = 1
   nvRAM_SPI_1_CS_Reg_Write(1);
}
```

10.7 SPI Processor Companion Read

```
/*****************PSoC3 Based Pseudo Code for SPI Processor Companion
Read***************/
uint8 READ_RTC (uint8 addr, uint8 *data_read_ptr, uint8 total_data_count)
{
   uint8 i;

   // Clear the Transmit Buffer
   nvRAM_SPI_1_SPIM_ClearTxBuffer();

   // Make chip select LOW CS = 0
   nvRAM_SPI_1_CS_Reg_Write(0);
   // Send Processor Companion Read Command
   nvRAM_SPI_1_SPIM_WriteTxData(nvRAM_RTC_READ_CMD);
   // Send Processor Companion Read Register Address
   nvRAM_SPI_1_SPIM_WriteTxData((uint8)(addr));
   // Wait for the transfer to complete
   while((nvRAM_SPI_1_SPIM_ReadTxStatus() & nvRAM_SPI_1_SPIM_STS_SPI_DONE) !=
   nvRAM_SPI_1_SPIM_STS_SPI_DONE);

   // Read the data and store in data_read_ptr
   for(i = 0; i < total_data_count; i++ )
   {
      // Clear receive buffer
      nvRAM_SPI_1_SPIM_ClearRxBuffer();
      // Send a dummy byte
      nvRAM_SPI_1_SPIM_WriteTxData((uint8)0x00);
      // Wait for the transfer to complete
      while((nvRAM_SPI_1_SPIM_ReadTxStatus() & nvRAM_SPI_1_SPIM_STS_SPI_DONE)
!=
      nvRAM_SPI_1_SPIM_STS_SPI_DONE);
      // Wait till the receive buffer has received a byte
     while(!nvRAM_SPI_1_SPIM_GetRxBufferSize());
      // Copy the read byte to data_read_ptr
      data_read_ptr[i] = nvRAM_SPI_1_SPIM_ReadRxData();
   }

   // Make chip select HIGH CS = 1
   nvRAM_SPI_1_CS_Reg_Write(1);
}
```

Document History

Document Title: AN408 - A Design Guide to SPI F-RAM™ Processor Companion - FM33256B

Document Number: 001-87564

Revision	ECN	Orig. of Change	Submission Date	Description of Change
**	4039506	MEDU	06/25/2013	New Spec.
*A	4571568	MEDU	11/17/2014	Added PSoC 3-based User Module project Replaced Pseudo codes with PSoC 3-based application code. Rewording / typo fixes
*B	5293268	MEDU	06/02/2016	Added a reference to code example CE204087 Updated template
*C	5803629	AESATMP9	07/07/2017	Updated logo and copyright.

Worldwide Sales and Design Support

Cypress maintains a worldwide network of offices, solution centers, manufacturer's representatives, and distributors. To find the office closest to you, visit us at Cypress Locations.

Products

ARM® Cortex® Microcontrollers	cypress.com/arm
Automotive	cypress.com/automotive
Clocks & Buffers	cypress.com/clocks
Interface	cypress.com/interface
Internet of Things	cypress.com/iot
Memory	cypress.com/memory
Microcontrollers	cypress.com/mcu
PSoC	cypress.com/psoc
Power Management ICs	cypress.com/pmic
Touch Sensing	cypress.com/touch
USB Controllers	cypress.com/usb
Wireless Connectivity	cypress.com/wireless

PSoC® Solutions

PSoC 1 | PSoC 3 | PSoC 4 | PSoC 5LP | PSoC 6

Cypress Developer Community

Forums | WICED IOT Forums | Projects | Videos | Blogs | Training | Components

Technical Support

cypress.com/support

Cypress Semiconductor
198 Champion Court
San Jose, CA 95134-1709