AN98553

S25FS-S Programming Guide

Author: Zhi Feng

Associated Part Family: S25FS-S

AN98553 describes the new features of the Cypress S25FS-S Flash device family.

1. Introduction

As a flash user, you may already be familiar with basic operations in SPI flash devices: Read, Program, and Erase. The S25FS-S Flash Family (S25FS512S, S25FS256S, and S25FS128S) provides many other features to satisfy the diverse needs of different users. This document is not intended to repeat the basic operations described in the data sheet, but to point out some important information in the data sheet that may be overlooked by users. Users should first seek to understand the data sheet before reading this document, especially if they are not familiar with Cypress SPI devices in general.

The document is designed for readers such as software developers who are writing low level drivers, set-up software, or application software for the S25FS-S devices.

2. FS-S Addressing Schemes

For devices that are 128 Mb or lower, address length of 3 bytes is sufficient to address the whole device. For devices that are of higher densities, address length of 4 bytes is needed. To accommodate these different address length requirements, the FS-S family devices provide two alternatives for users:

- A new set of commands that always require 4-byte address. These commands can be used to access up to 32 Gb of memory. The commands include all read commands, page program commands, erase commands and DYB/PPB protection commands.
- A 4-byte addressing mode for 3-byte address commands. This mode is controlled by the AL bit (Bit 7) in Configuration Register 2 (CR2). When this bit is set to 1, all standard 3-byte address commands require 4-byte addressing. The default of this bit is 0, i.e. 3-byte addressing scheme.

Note that when AL bit is set to 4-byte addressing mode, all standard 3-byte commands will require 4-byte addresses, except one command, RSFDP (5Ah). This command always uses 3-byte addressing scheme, regardless the current addressing mode as required by the JEDEC JESD216 (SFDP) standard.

Also note that while the S25FS128S devices do not need four-byte addressing, it supports the 4-byte address mode and commands. This allows driver software to be written to use 4-byte addressing with all of the family members so that migration between any of the densities is simplified by using the same software implementation with regard to address length.

3. FS-S Page Programming Size and Data Alignment

The FS-S family supports page programming with page size being either 256 or 512 bytes, depending on the value of Bit 4 of Configuration Register 3 (CR3). The page size is the upper limit of the size of data that can be entered in one program command. Users can actually program from one byte up to the maximum page size. If the data entered cross the page address boundary, the data will be wrapped back to the beginning of the page. Original data in the buffer could be overwritten.

Many applications store data in multiples of 512 bytes. Programming data to the flash is most efficient when writing in buffer-size length and aligned increments. Although smaller writes are allowed, software should be modified, whenever possible, to program data in full, address aligned, buffer increments.

For smaller or misaligned data writes, it is important to note that internally, data is programmed in address aligned groups of 16 bytes. For optimal flash performance and reliability, data should be programmed in multiples of full 16-byte aligned groups, up to the buffer size. While multiple program operations within a page are not recommended on FS-S devices, they are allowed for compatibility with legacy SPI devices.

For example, a simple flash file system may write two 512-byte file sectors, each with 12 bytes of metadata. Programming this data sequentially would cause several misalignments, as shown in Table 1.

Table 1 Misaligned Data Storage

Order	1st	2nd	3rd	4th
Size	512 bytes	12 bytes	512 bytes	12 bytes
Byte Offset	0	512	524	1036

Data Written	Initial 512 bytes			12 bytes	512 bytes				12 bytes	Not Written
Internal Groups	Group 0	…	Group 31	Group 32	Group 33	…	Group 63	Group 64	Group 65	Group 66

Instead, the writes should be rearranged to maximize programming performance. In Table 2, sector data is written from the bottom of flash, and metadata is written from the top. The S indicates four bytes that are skipped and left unused. Group N is the last group in the device. M is the size of the flash device in bytes.

Table 2 Aligned Data Storage

Order	1st	2nd	3rd	4th
Size	512 bytes	12 bytes	512 bytes	12 bytes
Byte Offset	0	M - 16	512	M - 32

Data Written	Initial 512 bytes			512 bytes			Not Written	12 bytes	S	12 bytes	S
Internal Groups	Group 0	…	Group 31	Group 32	...	Group 63	...	Group N-1		Group N	

4. FS-S Sector Erase Commands and Sector Architecture

The sector architecture in the FS-S family is very flexible. It provides both large “normal” sectors and small “parameter” sectors. Large sectors are 64 kB or 256 kB depending on the value of Bit 1 of the Configuration Register 3 (CR3). Parameter sectors are 4 kB in size. A small set of eight parameter sectors can be located at the lowest (bottom) or highest (top) address of a device. Parameter sectors can also be removed from the address space of the device so that all sectors are uniform in size.

To erase these two types of sectors, small parameter sectors and uniform size sectors, FS-S provides two sets of commands:

- Parameter 4-kB Erase. Two commands are provided: 20h (P4E) for use with 3- or 4-byte addressing; 21h (4P4E) for use with 4-byte addressing.
- Sector Erase. Two commands are provided: D8h (SE) for use with 3- or 4-byte addressing; DCh (4SE) for use with 4-byte addressing.

In order to erase parameter sectors, users need to issue P4E or 4P4E command. To erase uniform sectors, whether in 64 kB or 256 kB, users need to issue SE or 4SE command.

If a command is issued with the address of a sector size not matching the command, it may be ignored. This is discussed in more details below.

The following tables show all sector combinations an FS256S device may have.

Table 3 S25FS256S Sector Address Map, Bottom 4-kB Sectors, 64-kB Physical Sectors

Sector Size (kbyte)	Sector Count	Sector Number	Sector Address
4 kB	8	SA00	00000000h -- 00000FFFh
		:	:
		SA07	00007000h – 00007FFFh

Table 3 S25FS256S Sector Address Map, Bottom 4-kB Sectors, 64-kB Physical Sectors (Continued)

Sector Size (kbyte)	Sector Count	Sector Number	Sector Address
32 kB	1	SA08	00008000h – 0000FFFFh
64 kB	511	SA09	00010000h – 0001FFFFh
		:	:
		SA519	01FF0000h – 01FFFFFFh

Table 3 shows the bottom boot configuration, 8 small 4-kB parameter sectors are at the bottom of the device. As you can see, in addition to the bottom 4-kB sectors and regular 64-kB sectors, there is also a mid-size sector that has 32 kB. It is important to understand the reason for the mid-size is that the uniform sector size is still 64 kB. However, because the lowest address uniform sector is overlaid by the 8 small 4-kB sectors, the lower half of its memory cannot be accessed by the user, as illustrated in Figure 1. Therefore, the user accessible size for this sector is reduced to 32 kB. However, if the user is to execute a sector erase command, i.e. D8h or DCh, on an address ranging from 0000000h to 0000FFFFh, (full 64-kB range), the content of this mid-size sector will be erased, but the 4-kB parameter sectors will not be affected.

By issuing a 4-kB parameter sector erase command, i.e. 20h or 21h, on an address ranging from 00000000h to 00007FFFh, the corresponding 4-kB sector will be erased. If the command is issued on any other address space, the command will be ignored.

Figure .1 Mid-Size Sector

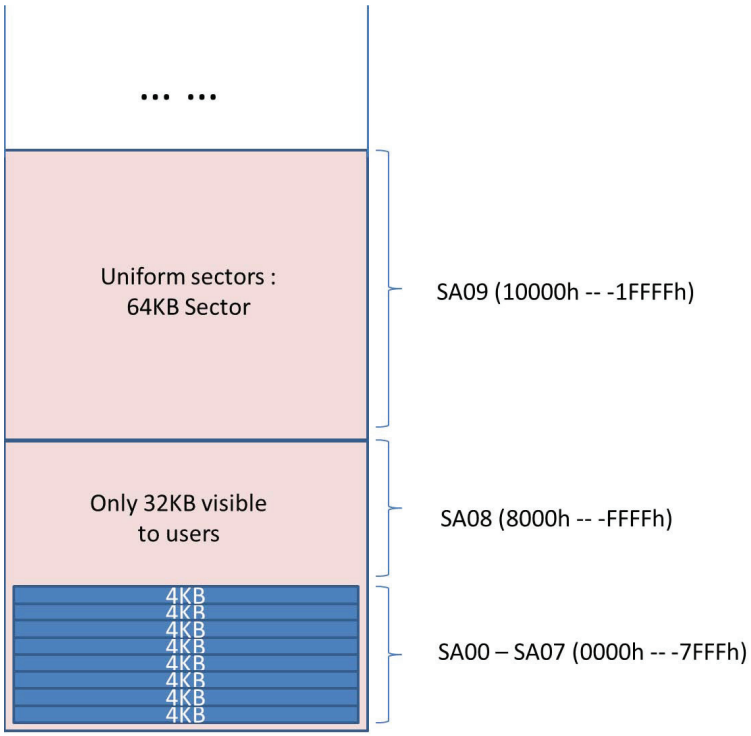

Table 4 shows the top boot configuration, 8 small 4-kB parameter sectors are at the top of the device. Similarly, if the user executes a sector erase command, i.e. D8h or DCh, on an address ranging from 01FF0000h to 01FFFFFFh, (full 64-kB range), the content of this mid-size sector will be erased, but the 4-kB parameter sectors will not be affected.

Table 4 S25FS256S Sector Address Map, Top 4-kB Sectors, 64-kB Physical Sectors

Sector Size (kbyte)	Sector Count	Sector Number	Sector Address
64 kB	511	SA00	00000000h -- 0000FFFFh
		:	:
		SA510	01FE0000h – 01EFFFFFh
32 kB	1	SA511	01FF0000h – 01FF7FFFh
4 kB	8	SA512	01FF8000h – 01FF8FFFh
		:	:
		SA519	01FFF000h – 01FFFFFFh

Table 5 shows a device with no small sectors, all in 64-kB size. There is no mid-size sector in this configuration because no memory is overlaid.

Table 5 S25FS256S Sector Address Map, 64-kB Uniform Sectors

Sector Size (kbyte)	Sector Count	Sector Number	Sector Address
64 kB	512	SA00	00000000h -- 0000FFFFh
		:	:
		SA511	01FF0000h – 01FFFFFFh

Table 6 shows the bottom boot configuration, 8 small 4-kB parameter sectors are at the bottom of the device and uniform sectors are 256 kB in size. You can observe that the mid-size sector is in between the small sectors and the regular sectors. Again, that is due to the 256-kB sector being overlaid by the 8 small sectors. It leads to the 224-kB size for the visible portion of this uniform sector. If the sector erase command, i.e. D8h or DCh, is issued anywhere at the bottom 256-kB address space (0-3FFFFh), the mid-size sector will be erased. The 4-kB parameter sectors will not be affected.

Table 6 S25FS256S Sector Address Map, Bottom 4-kB Sectors, 256-kB Physical Sectors

Sector Size (kbyte)	Sector Count	Sector Number	Sector Address
4 kB	8	SA00	00000000h -- 00000FFFh
		:	:
		SA07	00007000h – 00007FFFh
224 kB	1	SA08	00008000h – 0003FFFFh
256 kB	127	SA09	00040000h – 0007FFFFh
		:	:
		SA135	01FC0000h – 01FFFFFFh

Table 7 shows the top boot configuration, 8 small 4-kB parameter sectors are at the top of the device. If the user executes a sector erase command, i.e. D8h or DCh, on an address ranging from 01FC0000h to 01FFFFFFh, (full 256-kB range), the content of this mid-size sector will be erased. The 4-kB parameter sectors will not be affected.

Table 7 S25FS256S Sector Address Map, Top 4-kB Sectors, 256-kB Physical Sectors

Sector Size (kbyte)	Sector Count	Sector Number	Sector Address
256 kB	127	SA00	00000000h – 0003FFFFh
		:	:
		SA126	01F80000h – 01FBFFFFh
224 kB	1	SA127	01FC0000h – 01FF7FFFh
4 kB	8	SA128	01FF8000h – 01FF8FFFh
		:	:
		SA135	01FFF000h – 01FFFFFFh

Table 8 shows a device with no small sectors, all in 256-kB size. There is no mid-size sector in this configuration because no memory is overlaid.

Table 8 S25FS256S Sector Address Map, 256-kB Uniform Sectors

Sector Size (kbyte)	Sector Count	Sector Number	Sector Address
256 kB	128	SA00	00000000h – 0003FFFFh
		:	:
		SA127	01FC0000h – 01FFFFFFh

The above tables show the sector architectures of an FS256S device. Some of the configurations show a mid-size sector due to the memory overlay. A software developer will need to pay special attention to the Sector Erase command to make sure the address associated with the command is correct.

The same principle will apply to FS128S and FS512S devices. The only difference would be the total number of uniform sectors.

Note that in FS512S devices, there is no option for 64-kB uniform sectors. All uniform sector are 256 kB in size. As a result, the mid-size sector is always in 224 kB in size if applicable.

In FS-S family devices, users have an option to enable blank check feature during an erase. By default, when an erase command is issued, the sector is unconditionally erased. However, if the blank check feature is enabled, by turning on the Bit 5 of the Configuration Register 3 (CR3), the device will first check if the last erase was successfully completed on this sector (by the EES function described in the next section) and the sector is all blank. If so, it returns successful erase status. This dramatically reduces the erase time. If the blank check finds any 0 values in the array, the erase operation starts immediately.

This blank check feature is very useful, especially in the manufacturer environment where most of times the devices being programmed are brand new. But in order to make sure the successful programming, most of manufacturer software will perform an erase regardless. With the blank check feature enabled, the erase time will be improved dramatically when the device is new. And also it will erase the sector properly when it encounters a non-blank sector once a while.

5. Evaluate Erase Status (EES)

The FS-S family introduces a new command that allow the software to check whether or not an erase operation was interrupted by a power disruption, a hardware reset or a software reset. Users can first issue this command, D0h, with a particular sector address; then read the Bit 2 of the Status Register 2 (SR2) to check if the last erase on this sector was successfully completed.

Most Flash File Systems have software mechanisms to detect if an erase is interrupted by a power loss. In that case, it is necessary to re-erase the same sector to ensure the integrity of the flash array. With this new EES command, the File System software, or its block driver, can easily check the integrity of the sectors after each power up.

6. Status Registers (SRs) and Configuration Registers (CRs)

6.1 Access

The FS-S family provides many control and customization abilities to users, through a series of SRs and CRs. Most of these registers have two versions: non-volatile and volatile. The register value in a non-volatile register will be retained after a power cycle. The register value in a volatile register is reset back to the same value as its non-volatile counterpart.

The existence of a volatile version of the registers provides users the ability to test out settings in the early product development phase before programming the value to the non-volatile registers. Most of the bits in non-volatile registers are OTP (One Time Programmable) bits, thus the changes are not reversible. The volatile version of the registers also allows for overriding the value loaded during reset from the non-volatile register.

The detailed definitions of the SRs and CRs can be found in the data sheet. In this document, we would like to point out some useful information for software developers.

There are two general ways to access SRs and CRs:

1. Traditional way. These commands exist in older Cypress SPI devices:
   a. Using WRR (01h) to write to SR1NV, and CR1NV;
   b. Using RDSR1 (05h), RDSR2 (07h), or RDCR (35h) to read SR1V, SR2V or CR1V.

   Note that the WRR command writes to the non-volatile version of the registers; while the read commands read from the volatile version of the registers. When writing to the non-volatile registers, the volatile version will be updated automatically.
2. New commands. The FS-S family introduces a set of new commands to access any registers.
   a. WRAR (71h) to write to any register
   b. RDAR (65h) to read any register

   Note that to use these new commands, the register addresses in Table 9 should be used.

Table 9 Register Addresses for RDAR and WRAR Commands

Byte Address (Hex)	Register Name
00000000	SR1NV
00000001	NA
00000002	CR1NV
00000003	CR2NV
00000004	CR3NV
00000005	CR4NV
...	NA
00000010	NVDLR
...	NA
00000020	PASS[7:0]
00000021	PASS[15:8]
00000022	PASS[23:16]
00000023	PASS[31:24]
00000024	PASS[39:32]
00000025	PASS[47:40]
00000026	PASS[55:48]
00000027	PASS[63:568]
...	NA
00000030	ASPR[7:0]
00000031	ASPR[15:8]
...	NA
00800000	SR1V
00800001	SR2V
00800002	CR1V
00800003	CR2V
00800004	CR3V
00800005	CR4V
...	NA
00800010	VDLR
...	NA
00800040	PPBL
...	NA

6.2 Order of Execution

Some bits in CRs are important to the sector architecture of the device. It is necessary to have these register bits set before any program or erase is done to the device. These bits are:

- TBPARM (Bit 2) in CR1NV: This bit determines where the parameter sectors are, i.e. bottom or top.
- 20h_NV (Bit 3) in CR3NV: This bit determines if parameter sectors exist in user memory.
- D8h_NV (Bit 1) in CR3NV: This bit determines the size of uniform sectors, i.e. 64 kB or 256 kB.

If the user modifies any of these bits after some program operations to the main array, the contents of the array are not guaranteed to still be there. Therefore, for the best practice, users should configure all these CR bits before accessing the flash array.

Some SR and CR bits can be modified with the same command but some bits have protection interactions with each other. For example, the WRR command can write both SR1 and CR1 in one command. The BP bits are in SR1 and the FREEZE bit is in CR1V. It is recommended in software to set first the BP bits, then use a separate WRR command to set FREEZE bit to protect the BP bits. However, if the user issues the new BP bit values and FREEZE bit value in the same command, it will still work because the device will act upon the current FREEZE value.

Once the user chooses a protection mode for the device, i.e. Persistent Protection or Password Protection (the protection modes will be discussed later on in this document), CR1NV (except FREEZE and QUAD), CR2NV, CR3NV and CR4NV are protected. It is important that if any modification is needed in these registers, it must be done before choosing the protection mode.

7. Protection

7.1 BP Bit Protection

The BP bit protection in the FS-S family works exactly the same as old Cypress SPI devices. The BP bits protect part or all of the flash memory depending on the values.

There are two versions of the BP bits. The Non-volatile version is in SR1NV. The Volatile version is in SR1V. When using RDSR (05h) command to read, it always reads the SR1V value. If the user wants to read the non-volatile version of the BP bits, the RDAR (65h) command should be used. The SR1NV value will be returned.

When using the WRR (01h) or WRAR (71h) command to write BP bits, depending on the BPNV value (in CR1V), the command writes to the BP bits in SR1NV or SR1V.

7.2 Advanced Sector Protection (ASP) Protection

There are two ASP modes in the FS-S family, Persistent Protection mode and Password Protection mode. Users can select one by programing Bit 1 or Bit 2 of the ASP Register. Note that these two bits are mutually exclusive. If a user tries to program both bits to 0, the program command will result in an error and neither bit will be changed.

Once one of the modes is selected, most of the bits in CRs are protected, as mentioned earlier in this document. So it is important to program all the CRs first before choosing the protection mode.

If the protection mode has not been selected, the device will function as if in Persistent Protection mode. Cypress strongly recommend that users explicitly select the desired mode so that malicious code can not later change the protection behavior of the device.

8. Multiple Input / Output (MIO)

8.1 Dual and Quad Output Commands

The S25FS064S supports Dual Output Read (DOR/4DOR) and Quad Output Read (QOR/4QOR) commands like previously released Cypress SPI device families. S25FS128S, S25FS256S and S25FS512S do not support Dual and Quad Output commands.

8.2 Dual and Quad Input/Output (I/O) Commands

The all S25FS-S family devices support Dual and Quad Input/Output (I/O) commands just like previously released Cypress SPI device families.

8.3 Quad Page Program Command

The S25FS064S supports Quad Page Program (QPP/4QPP) command like previously released Cypress SPI device families. S25FS128S, S25FS256S and S25FS512S do not support Quad Page Program command.

8.4 Quad Peripheral Interface (QPI) Mode

The all S25FS-S family devices support QPI mode in which all information, including the instruction code, is transferred in 4-bit width. In the data sheet, this is referenced as 4-4-4 command protocol, as the instruction, address and data are all transferred in 4-bit width.

To operate in QPI mode, users write to the QA (Bit 6) bit in CR2V or CR2NV. Note that the QA bit in CR2NV is an OTP bit; once the user programs the bit to 1 the device always operates in QA mode after reset.

This is a non-reversible operation.

If the user wants to try out the QPI mode, it is suggested to use QA_V bit in the CR2V register. This is the volatile version of the QA bit and the device returns to normal mode after reset. Note that once QA_V bit is set, the Quad bit (Bit 1) in CR1V is set automatically. That indicates all IO signals are used for information transfer and the WP# and HOLD# functions are disabled. When QA_V bit is reset back to 0, the Quad Bit in CR1V will remains 1. The user needs to reset it back to 0 if necessary. Once the QPI mode has been tested thoroughly, the user can set QA_NV bit to permanently run the device in QPI mode if desired. Although with QA_NV bit set, the user can still set QA_V bit to 0 so the device reverts back to normal mode, this is not a recommended operation because the device would always revert back to QA after reset.

9. Secured Silicon Region (SSR)

The FS-S family provides 1024 bytes of One Time Program (OTP) area separated from the main flash array. This area is also called SSR. The area is divided into thirty two, individually lockable, 32-byte aligned regions. (32 x 32 bytes = 1024 bytes)

When reading from the SSR, if the address entered is outside of the 1024-byte region, or a read operation extends beyond the 1024-byte region, the read data will be undefined.

When programming to the SSR, if the address entered is outside of the 1024-byte area, the program command will be ignored. No error is reported.

The SSR is protected by the FREEZE bit in CR1V. If FREEZE is set, the SSR program command will be ignored. No error is reported.

The Region 0 of the SSR (first 32 bytes) is a special region. The first 16 bytes of Region 0 is reserved for Cypress to program in a Random Number that can be used as a unique device identification e.g. serial number. The next 4 bytes are the Lock Bits. Each lock bit controls the corresponding 32 SSR regions, from Region 0 to Region 31.

Any attempt to program to the Random Number area will result in a program error.

If an SSR region is locked by its lock bit, any attempt to program into the region will result in a program error.

When programming the SSR, the program page size is the same as the normal flash array page program, i.e. either 256 or 512 bytes depending on the 02H_V bit in CR3V. That means the user can program multiple SSR regions in the same program command.

If the program data entered is more than the page size, the data will be wrapped to the beginning of the page, just like a normal page programming command. In this case, the wrapped data may coincide with Region 0, which is a special region as mentioned above. In this case, the program command may fail if it contains some data intended for the first 16 bytes of Region 0. Loading data beyond the end of the page programming buffer is not recommended.

10. Allowed Commands in Suspend

The FS-S family supports both erase suspend and program suspend. The suspend function is typically used in embedded systems where the CPU needs to read part of the flash before an erase or program completes, to guarantee an acceptable system response latency.

For backward and alternate source compatibility reasons, there are three erase/program suspend commands users can choose: 75h, B0h, and 85h. These commands all function the same way. There are also three erase/program resume commands: 7Ah, 8Ah and 30h. The first two behave the same way. The 30h command can be used for CLSR (Clear Status Register) or Erase/Program Resume, depending on 30h_V bit in CR3V.

When the device is in erase suspend or program suspend state, it supports only a subset of commands. As a software developer, one should not try to execute other commands if the device is in suspend state. The following table summarizes what commands are supported:

Table 10 Commands Supported in Suspend States

Command Name	Hex Value	Supported in Erase Suspend	Supported in Program Suspend
PP and 4PP [1]	02h and 12h	Yes	
READ and 4READ	03h and 13h	Yes	Yes
FAST_READ and 4FAST_READ	0Bh and 0Ch	Yes	Yes
DIOR and 4DIOR [1]	BBh and BCh	Yes	Yes
QIOR and 4QIOR [1]	EBh and ECh	Yes	Yes
DDRQIOR and 4DDRQIOR	EDh and EEh	Yes	Yes
MBR	F0h	Yes	Yes
RDSR1	05h	Yes	Yes
RDSR2	07h	Yes	Yes
RDAR	65h	Yes	Yes
WREN	06h	Yes	
CLSR	30h or 82h	Yes	
EPR	7Ah, 8Ah or 30h	Yes	
RSTEN	66h	Yes	Yes
RST	99h	Yes	Yes
RESET	F0h	Yes	Yes
DYBRD and 4DYBRD	FAh and E0h	Yes	
DYBWR and 4DYBWR	FBh and E1h	Yes	
PPBRD and 4PPBRD	FCh and E2h	Yes	

Note
1. S25FS064S only

11. Summary

This guide is intended as a supplement to the FS-S data sheet to further help software developers to better design their low level drivers and application software. It describes some important functions in the FS-S family and how to configure and operate the devices.

Document History Page

Document Title: AN98553 - S25FS-S Programming Guide Document Number: 001-98553				
Rev.	**ECN No.**	**Orig. of Change**	**Submission Date**	**Description of Change**
**	–	–	01/16/2013	Initial release
*A	4922312	ZHFE	09/28/2015	Updated in Cypress template
*B	5885154	AESATMP8	09/15/2017	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 | Video | Blogs | Training | Components

Technical Support

cypress.com/support

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