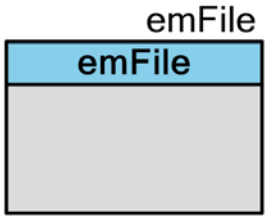
文件系统库 (emFile)

1.0



特性

- 多达四个 SPI 模式安全数字 (SD) 卡
- FAT12/16 或 FAT32 格式
- 可以选择与操作系统 (OS) 集成
- 可选长文件名 (LFN) 处理

概述

EmiFile 组件提供了与具有 FAT 文件系统格式化的 SD 卡通讯的接口。SD 卡规范包含有与 一个 SD 卡通信的多个硬件接口。此组件使用 SPI 接口方式进行通信。可以将最多四个独立 SPI 接口用于与每个 SD 卡通信。同时支持 FAT12/16 和 FAT32 文件系统格式。此组件提供与 SD 卡之间的物理接口，使用 SEGGER Microcontroller 授权的 emFile 库提供函数库，以操控 FAT 文件系统。

何时使用 emFile

使用 emFile 组件可访问使用 FAT 12/16 或 FAT32 文件系统格式进行格式化的 SD 卡。

入门

安装 emFile 库

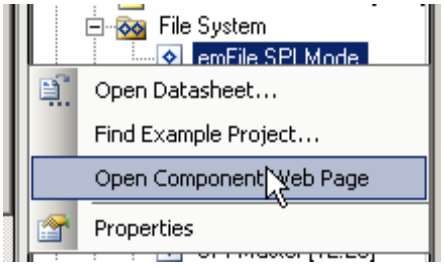
emFile 文件系统实现包含两个部分。第一部分是随 PSoC Creator 附带的 emFile 组件。第二部分是 SEGGER Microcontroller 授权的 emFile 文件系统库。该库以 zip 文件形式提供，可以从赛普拉斯网站 emFile 组件页面下载。

安装 emFile 库：

1. 打开 PSoC Creator 并转到 Component Catalog（组件目录）窗口。搜索 emFile 组件。该组件位于 （赛普拉斯）选项卡下的 **Communications**（通信）> **File System**（文件系统）> **emFile SPI Mode**（emFile SPI 模式）
2. 右键单击 emFile 组件。

   显示菜单如图一所示。

   图 1. emFile Open Component Web Page（打开组件网页）

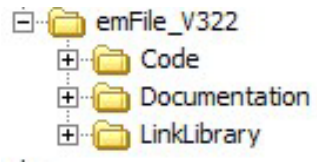
3. 单击 **Open Component Web Page**（打开组件网页）。

   进入登陆页面，页面里包含最新的文件系统，文件的格式为 zip。
4. 下载该 zip 文件并将其提取到所选的文件夹下（保持该 zip 文件的目录结构）。确保文件在提取之后不是只读格式。

emFile 库目录组织

解压缩的 emFile 库的目录结构类似于图 2 所示。

图 2. emFile 目录组织



Code（代码）目录包含以源代码格式提供的部分库。此目录具有两个子目录：Include（包括）和 Source（源代码）。Include（包括）目录提供库的头文件。Source（源代码）目录提供库的可执行部分，此库为源代码格式。

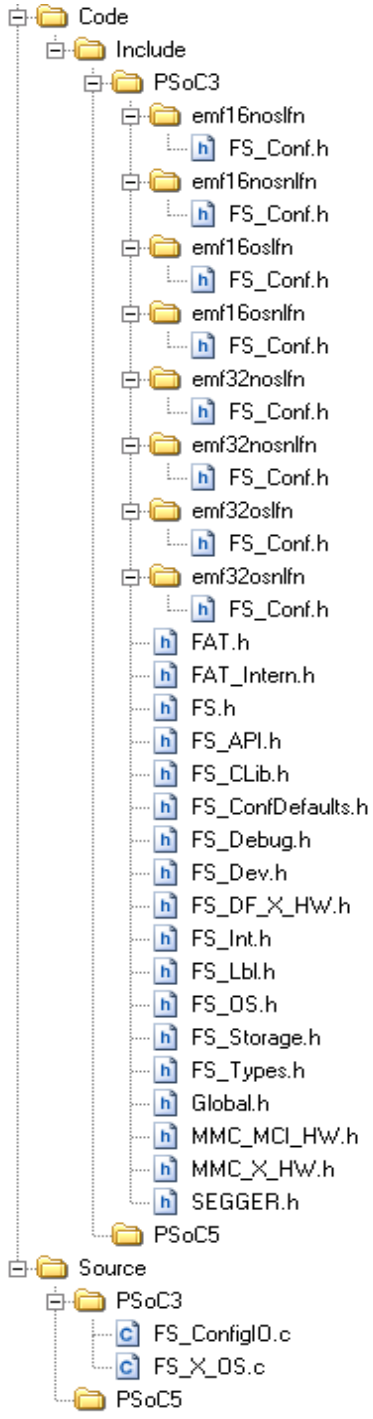
Include（包括）部分包含两部分，第一，始终应用于目录顶层的头文件第二，为特定 LinkLibary 选择的选项而使用的子目录中的头文件。子目录的名称指定为 emf<选项>。表 1 中介绍了所有选项。

表 1. 选项

选项	说明
16	FAT 12/16 格式
32	FAT 32 格式
Os	操作系统支持
Nos	无操作系统支持
Lfn	长文件名支持
Nlfn	无长文件名支持

图 3 显示 Code（代码）目录中的文件。此图显示 PSoC 3 的文件的目录组织。PSoC 5 具有类似文件组织。

图 3. emFile “Code（代码）”目录文件列表

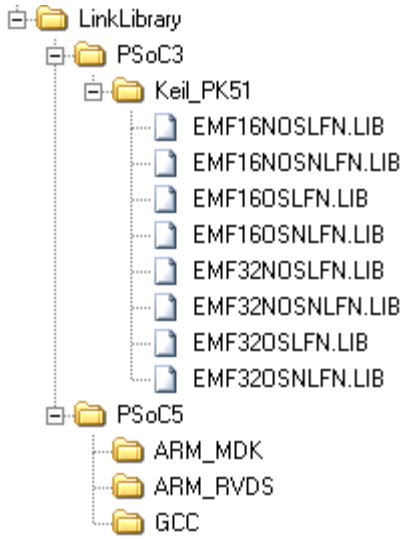


Documentation（文档）部分包含 SEGGER Microcontroller emFile 文件系统库的用户指南。

LinkLibrary 部分划分为 PSoC 3 和 PSoC 5 两个目录。每个目录包含用于每个被支持工具链的子目录。这些目录是为所选特定选项提供文件系统实现的对象库。使用本数据手册后面介绍的 Build

Settings（构建设置）选项，该库被添加到项目中。扩展的 PSoC 3/Keil PK51 部分包含图 4 中显示的对象库。

图 4. LinkLibrary 结构



LinkLibrary 的命名约定为：<前缀>emf<选项>.<扩展名>。选项与用于包括文件目录名称的选项相同前缀和扩展名应用于特定的工具链。请参见表 2。

表 2. 命名约定

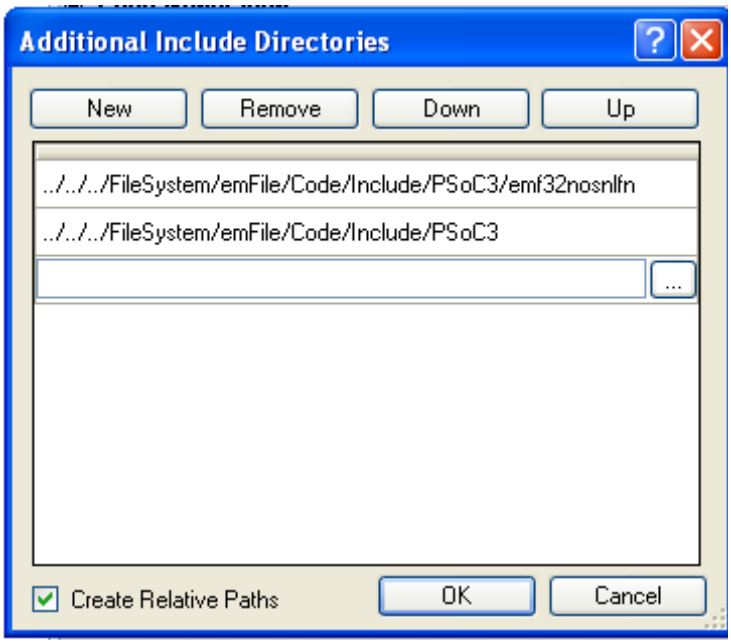
工具链	前缀	扩展名
Keil PK51		lib
GCC	lib	a
ARM_MDK		lib
ARM_RVDS		lib

为 PSoC 3 应用程序创建 emFile 项目

在 PSoC 3 项目中调用 emFile 库：

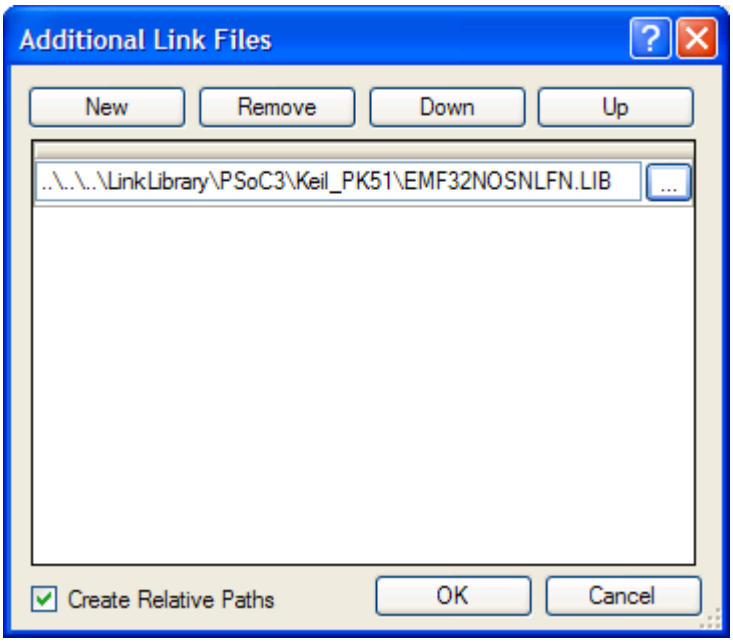
1. 选择所需的库。此操作基于是否需要 FAT12/16 还是 FAT32、应用程序是否使用操作系统以及是否需要长文件名支持。此示例使用 emf32nosnlfn.lib（FAT32、无操作系统以及无长文件名支持）。
2. 选择所需包括文件目录。转到 **Project**（项目）**> BuildSettings**（构建设置）**> Compiler**（编译器）**> General**（常规）。单击 **Additional Include Directories**（其他包括目录）属性字段中的“**…**”按钮。会显示 **Additional Include Directories**（其他包括目录）对话框。单击 **New**（新建）按钮，然后为 PSoC 3 选择目录并为所需特定选项选择目录。完成时，对话框如图 5 所示。

图 5. 添加包括目录



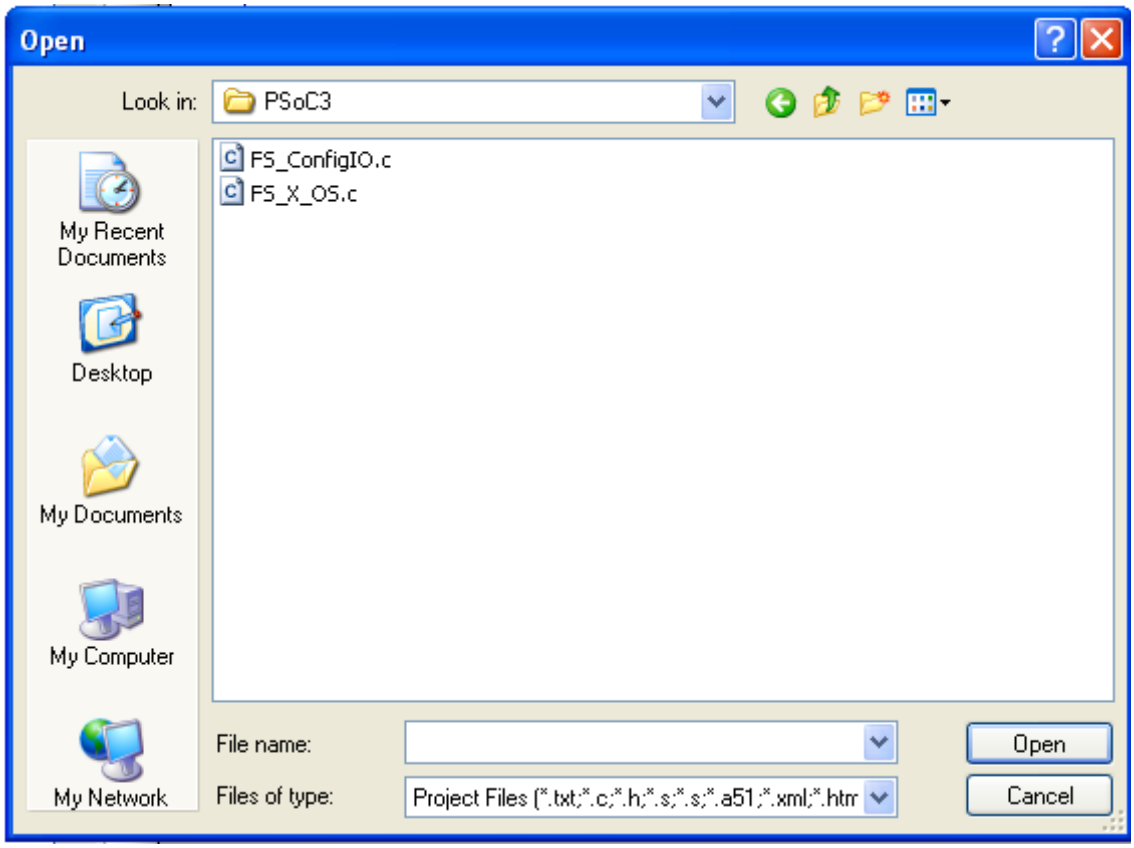
3. 选择所需链接库文件。转到 **Project**（项目）**> BuildSettings**（构建设置）**> Linker**（连接器）**> General**（常规）。单击 **Additional Link Files**（其他链接文件）属性字段中的“**…**”按钮。会显示 **Additional Link Files**（其他链接文件）对话框。单击 **New**（新建）按钮，并基于所需特定选项选择库文件。完成时，对话框如图 6 所示。

图 6. 添加链接库



4. 将源文件 *FS_ConfigIO.c* 添加到项目中。如果使用操作系统，还应添加 *FS_X_OS.c*。这些文件可以从 Code（代码）/Source（源代码）/PSoC3 目录直接添加到项目中，或先复制到项目目录下，然后从其中添加。这些文件通常将被编辑，因此需要决定是更改原始文件还是特定于项目的副本。这些文件使用 ***Project***（*项目*）***> Existing Item***（*现有项*）添加到项目中。会打开图 7 中显示的对话框，以便您可以选择文件。

图 7. 添加源文件



emFile 库现包含在项目中。

为 PSoC 5 应用程序创建 emFile 项目

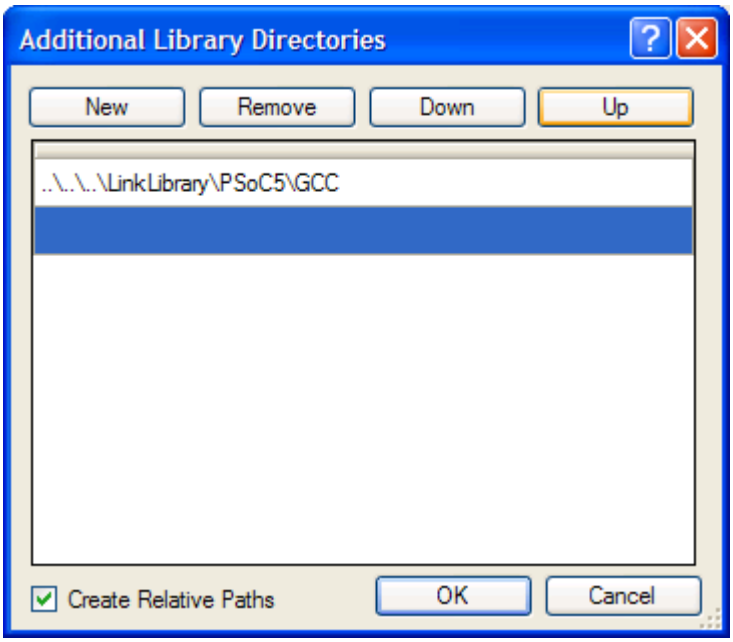
用于为 PSoC 5 创建 emFile 项目的步骤与用于 PSoC 3 的步骤相同（除了在使用 GCC 工具链时）。使用 GCC 工具链时，必须指定库文件所处的目录，而非以文件形式添加链接库。

转到 **BuildSettings**（构建设置）**> Linker**（连接器）**> General**（常规）**> Additional Library Directories**（其他库目录）。单击 **Additional Library Directories**（其他库目录）属性字段中的“**…**”按钮。

会显示 **Additional Library Directories**（其他库目录）对话框。

单击 **New**（新建）按钮并为 GCC 库选择库目录。完成时，对话框如图 8 所示。

图 8. 添加库目录



在库目录中指定库。

a. 转到 **BuildSettings**（构建设置）**> Linker**（连接器）**> General**（常规）**> Additional Libraries**（其他库）。

b. 键入库名称（从名称中排除“lib”前缀和“.a”后缀）。

   假设使用 libemf32nosnlfn.a 库文件，则库名称应为 emf32nosnlfn。

输入/输出连接

本节介绍 emFile 组件的各种输入和输出连接。I/O 列表中的星号 (*) 表示该 I/O 可能在 I/O 说明中列出的情况下从组件中排除。

符号上没有用于组件的可见连接。显示的所有连接用于组件内部包含的引脚连接。这些引脚都会显示在设计范围资源引脚编辑器中。必须使用引脚编辑器将它们分配给合适的物理引脚。对于每个连接，有四个引脚，分别使用索引 0 到 3 命名。这些引脚表示组件可以支持的四个独立 SD 卡。这四个引脚中的一个会出现在设计中，具体取决于所选 SD 卡数。下面是这些引脚的说明。

SPI0_WP – SPI3_WP*

可选输入引脚，基于 **SD Card [0-3] Write Protection**（SD 卡 [0-3] 写保护）选项，这些选项配置 SD 卡 [0-3] 的写保护。如果选中这些选项，则存在这些引脚。这些引脚的默认状态不存在，这表示不对 SD 卡进行写保护。

mosi0 – mosi3

SPI 主控的主出从入引脚。此引脚应连接到 SD 卡的 DI/CMD 引脚。

miso0 – miso3

SPI 主控的主入从出引脚。此引脚应连接到 SD 卡的 DO/DAT0 引脚。

sclk0 – sclk3

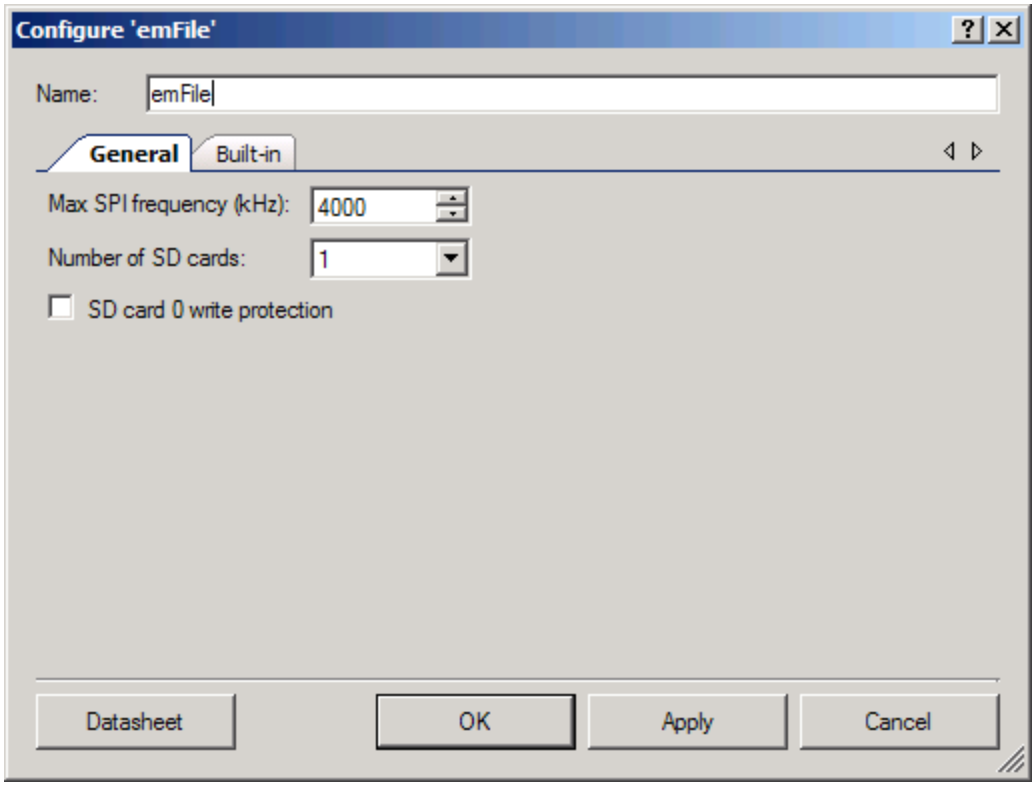
SPI 主控的串行时钟引脚。此引脚应连接到 SD 卡的时钟引脚。

SPI0_CS – SPI3_CS

卡选择输出引脚。此引脚应连接到 SD 卡的 nCS 引脚。

元件参数

将 emFile 拖动到您的设计中，双击打开 Configure（配置）对话框。



emFile 提供以下参数。

最大 SPI 频率

定义 SPI 主控组件对 SD 卡计时的最大频率。该值以 kHz 为单位。取值范围为 400 到 25,000 kHz，默认设置为 **4000**。

频率范围基于 SD 卡规范。请注意，对于 emFile 中使用的 SPI 主控 (v2.20) 组件的最大频率有所限制。有关此组件可以支持的最大频率的详细信息，请参考 SPI 主控 v2.20 数据手册“时序特性”。

SD 卡数

定义 emFile 文件中的 SD 卡数。该值可以设置为 1 到 4。默认设置为 **1**。

SD 卡 *n* 写保护

定义为每个 *n* SD 卡启用的写保护。它默认为禁用。

放置

emFile 组件放置于整个 UDB 阵列中，并且所有放置信息通过 *cyfitter.h* 文件提供给 API。

性能和资源使用情况

emFile 的性能取决于一组参数（CPU、编译器、优化、负载数据大小），并且受 SPI 主控组件的最大速度限制。下表包含读/写速度值（取决于各种影响 emFile 组件性能的因素）。

器件	CPU 速度	SPI 时钟	模式	性能 (KBps)							
				1 字节		256 个字节		1 KB		8 KB	
				写	读	写	读	写	读	写	读
PSoC 3	24 MHz	8 MHz	释放	0.02	0.14	6.56	28.44	22.73	76.92	32.26	68.96
PSoC 3	48 MHz	8 MHz	释放	0.04	0.26	10.93	49.23	38.46	119.0	54.05	117.6
PSoC 3	24 MHz	4 MHz	释放	0.02	0.13	5.98	25.6	21.28	66.67	30.07	64.52
PSoC 3	48 MHz	4 MHz	释放	0.04	0.21	9.55	42.2	32.47	108.7	45.44	100
PSoC 5	24 MHz	8 MHz	释放	0.05	0.41	13.61	106.6	47.17	208.3	87.92	205.12
PSoC 5	48 MHz	8 MHz	释放	0.06	0.5	16.2	142.2	55.56	250	103.92	242.4
PSoC 5	24 MHz	4 MHz	释放	0.04	0.29	9.99	75.29	33.78	138.9	62.96	145.44
PSoC 5	48 MHz	4 MHz	释放	0.05	0.38	12.59	98.45	42.37	192.3	80	186.08

存储器使用情况

emFile 组件存储器使用情况因应用而异。SEGGER 的文档提供了有关如何计算存储器资源的详细说明。下表包含文件系统一些常用功能的存储器使用情况值。所有值都以字节为单位。

emFile 模块	Keil_PK51				GCC-4.4.1			
	释放		调试		释放		调试	
	闪存	SRAM	闪存	SRAM	闪存	SRAM	闪存	SRAM
文件系统内核（SPI 驱动器）	28035	4762	28799	4849	10440	4272	12432	4272
读取文件	2668	6	2672	3	824	0	832	0
写入文件	1927	0	1932	0	808	0	816	0
打开目录	2714	47	2733	47	408	0	416	0
创建目录	898	0	898	0	464	0	480	0
删除文件	75	0	75	0	48	0	48	0
长文件名支持*	-	-	-	-	2232	0	2232	0
低级格式	769	0	769	0	240	0	232	0
格式化 SD 卡	6063	0	6063	0	2296	0	2304	0

* 长文件名支持代码会自动包含在 PSoC 3 的文件系统内核中。

应用程序编程接口

应用程序编程接口 (API) 子程序允许您使用软件配置组件。下表列出了每个函数的接口，并进行了说明。以下各节将更详细地介绍每个函数。

默认情况下，PSoC Creator 将给设计中的第一个组件命名为“emFile_1” 可以在遵循标识符语法规则的条件下，对其重新命名。实例名称会成为每个全局函数名称、变量和常量符号的前缀。出于可读性考虑，下表中使用的实例名称为“emFile”。

emFile 库子程序会自动初始化并启用 emFile 组件。为 emFile 组件提供的独有 API 旨在支持功耗模式之间的切换。

本节不包括 emFile 库的文件系统 API 的说明，因为在 *emFile User Guide* (《emFile 用户指南》) 的第 4 章中介绍了这些 API。必须将 *FS.h* 头文件包含在 *main.c* 文件中才能使用文件系统 API 。

注：若要在 PSoC 5 器件上使用长文件名 (LFN) 支持，必须调用 FS_FAT_SupportLFN()。对于 PSoC 3 器件，此功能在默认情况下是启用的。

函数	说明
emFile_Sleep()	准备 emFile 以进入睡眠模式。
emFile_Wakeup()	在退出睡眠模式之后恢复 emFile。
emFile_SaveConfig()	保存 HW 驱动器使用的 SPI 主控配置。
emFile_RestoreConfig()	恢复 HW 驱动器使用的 SPI 主控配置。

void emFile_Sleep(void)

说明： 准备 emFile 以进入睡眠。

参数： 无

返回值： 无

副作用： 无

void emFile_Wakeup(void)

说明： 在退出睡眠模式之后恢复 emFile。

参数： 无

返回值： 无

副作用： 调用 emFile_Wakeup() 函数前未调用 emFile_Sleep() 或 emFile_SaveConfig() 函数可能会产生意外行为。

void emFile_SaveConfig(void)

说明： 保存 HW 驱动器使用的 SPI 主控配置。此函数由 emFile_Sleep() 调用。

参数： 无

返回值： 无

副作用： 无

void emFile_RestoreConfig(void)

说明： 恢复 HW 驱动器使用的 SPI 主控配置。此函数由 emFile_Wakeup() 调用。

参数： 无

返回值： 无

副作用： 调用该函数前未调用 emFile_Sleep() 或 emFile_SaveConfig() 函数可能会产生意外行为。

固件源代码示例

PSoC Creator 在 Find Example Project（查找示例项目）对话框中提供了许多包括原理图和示例代码的示例项目。要获取组件特定的示例，请打开组件目录中的对话框或原理图中的组件实例。要获取通用的示例，请打开开始页或 **File**（文件）菜单中的对话框。根据需要，使用对话框中的 **Filter Options**（滤波器选项）可缩小可选项目的列表。

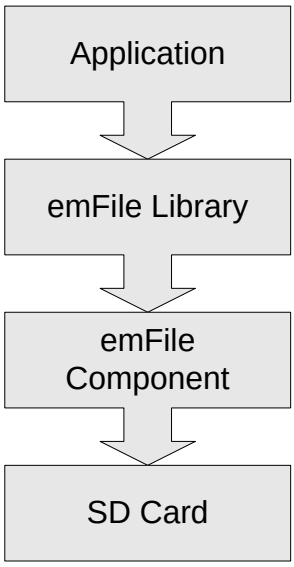
有关更多信息，请参考 PSoC Creator 帮助中的“查找示例项目”主题。

功能描述

文件系统库结构

emFile 文件系统实现由两个部分组成：emFile 组件和 SEGGER Microcontroller 授权的 emFile 文件系统库。文件系统应用程序会使用 emFile 库中提供的 API。该库会使用 emFile 组件提供以使用 SPI 接口的 SD 卡之间的物理接口。emFile 文件系统结构如图 9 所示。

图 9. emFile 结构



组件更改

版本 1.0 为 emFile 组件的首次发行版。

© 赛普拉斯半导体公司，2012。此处所包含的信息可能会随时更改，恕不另行通知。除赛普拉斯产品的内嵌电路之外，赛普拉斯半导体公司不对任何其他电路的使用承担任何责任。也不根据专利或其他权利以明示或暗示的方式授予任何许可。除非与赛普拉斯签订明确的书面协议，否则赛普拉斯产品不保证能够用于或适用于医疗、生命支持、救生、关键控制或安全应用领域。此外，对于可能发生运转异常和故障并对用户造成严重伤害的生命支持系统，赛普拉斯不授权将其产品用作此类系统的关键组件。若将赛普拉斯产品用于生命支持系统中，则表示制造商将承担因此类使用而招致的所有风险，并确保赛普拉斯免于因此而受到任何指控。

PSoC® 和 CapSense® 是赛普拉斯半导体公司的注册商标；SmartSense™、PSoC Creator™ 和 Programmable System-on-Chip™ 是赛普拉斯半导体公司的商标。此处引用的所有其他商标或注册商标归其各自所有者所有。

所有源代码（软件和/或固件）均归赛普拉斯半导体公司（赛普拉斯）所有，并受全球专利法规（美国和美国以外的专利法规）、美国版权法以及国际条约规定的保护和约束。赛普拉斯据此向获许可者授予适用于个人的、非独占性、不可转让的许可，用以复制、使用、修改、创建赛普拉斯源代码的派生作品、编译赛普拉斯源代码和派生作品，并且其目的只能是创建自定义软件和/或固件，以支持获许可者仅将其获得的产品依照适用协议规定的方式与赛普拉斯集成电路配合使用。除上述指定的用途之外，未经赛普拉斯的明确书面许可，不得对此类源代码进行任何复制、修改、转换、编译或演示。

免责声明：赛普拉斯不针对此材料提供任何类型的明示或暗示保证，包括（但不仅限于）针对特定用途的适销性和适用性的暗示保证。赛普拉斯保留在不做出通知的情况下对此处所述材料进行更改的权利。赛普拉斯不对此处所述之任何产品或电路的应用或使用承担任何责任。对于可能发生运转异常和故障并对用户造成严重伤害的生命支持系统，赛普拉斯不授权将其产品用作此类系统的关键组件。若将赛普拉斯产品用于生命支持系统中，则表示制造商将承担因此类使用而招致的所有风险，并确保赛普拉斯免于因此而受到任何指控。

产品使用可能受适用的赛普拉斯软件许可协议限制。