



Application Note

SPC1068 Flash 存储寿命延长使用指南

Revision 1 – October 2017

目录

1	引言	5
2	调用 API 进行存储.....	7
2.1	Data Manage 存储功能	7
2.2	Data Manage 读取功能	7
3	修订记录	9

表格列表

表 3-1. 文档修订记录	9
---------------------	---

图片列表

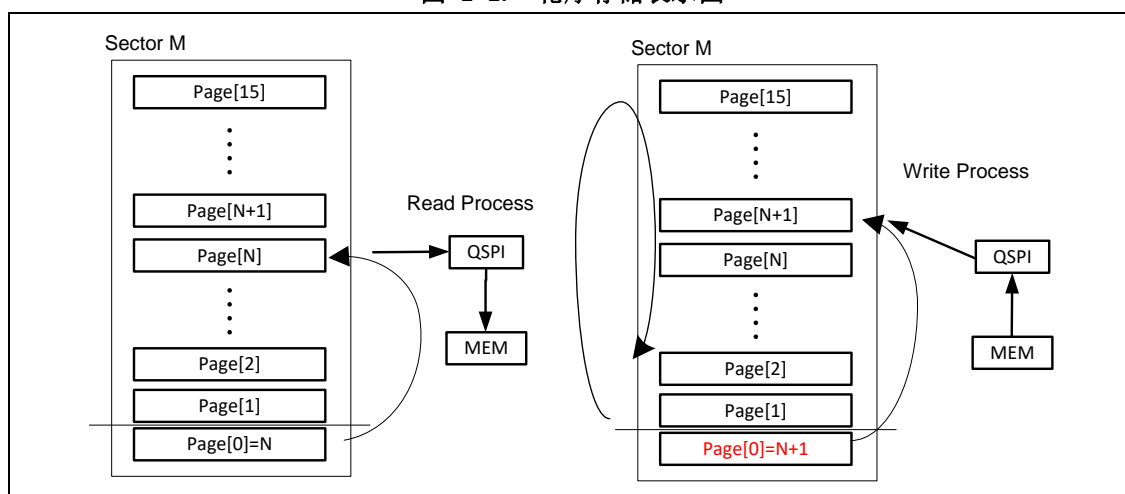
图 1-1.	轮序存储表示图	5
图 1-2.	提升擦写次数表示图	6
图 1-3.	提升单位存储量表示图	6
图 2-1.	设定存储位置	7

1 引言

SPC1068 内部集成了 SPI Flash，共有 512KB 的数据空间，每个 page(256 byte)可经历 100K 次的擦写，透过 Spintrol 提供之 data_manage 档案管理，可将档案存储寿命延长为 15 倍，亦即最少可支持 1500K(150 万次)次的擦写。

本技巧支持 256 byte 以内的数据量，使用 sector 内的 16 个 page(每个 page 256 byte)作为依序存储空间，减少擦写次数，使用原理如下

图 1-1. 轮序存储表示图

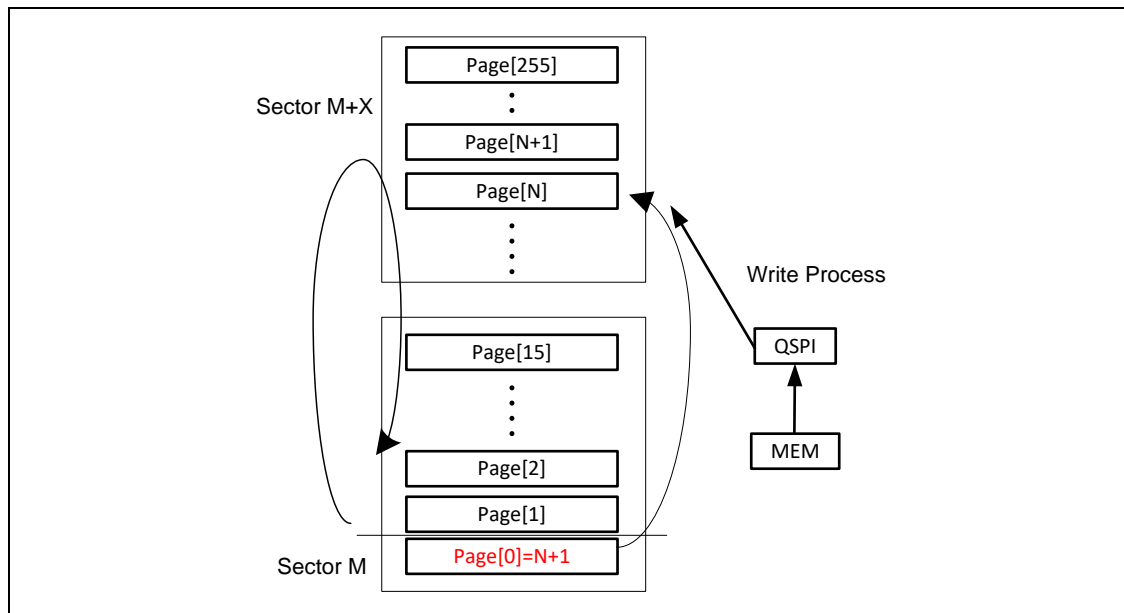


如上图，Page0 内会记录目前使用的 page 指针，每次存储前，会将此指标+1(由此 Page 内 1 翻 0 的个数代表指标数值，如 0xff...fe 代表 N=1，亦可大幅减少操作此 Page 的抹除数)，并且将数据存入相对 page 内。读取前，会读取此指标。如此一来便可以大幅减少 Flash 的擦写次数。

Spintrol 所提供之 Data manage 代码不保证数据写入前后的正确性，若使用者有需求，请自行加入 checksum 机制确保数据之正确性。

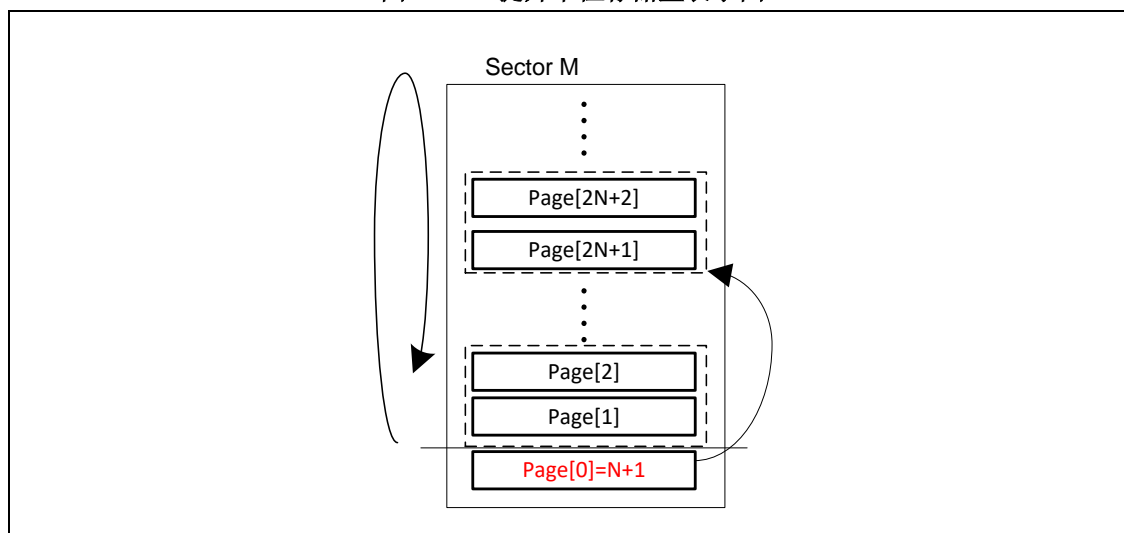
以上之方式之范例为 256 byte 之信息, 并且使用该 Sector 作为存储, 若欲突破 150 万次之擦写, 可以将指针延伸到下一个 Sector, Page0 最多可支持计数 255 次, 因此以 256 byte 来说最高抹除次数可提升至 3000 万次左右, 如下图所示。

图 1-2. 提升擦写次数表示图



若单一存储量大于 256 byte, 可以整并两个以上的 Page 作为一个存储单元, 用户可自行修改代码。如下图所示

图 1-3. 提升单位存储量表示图

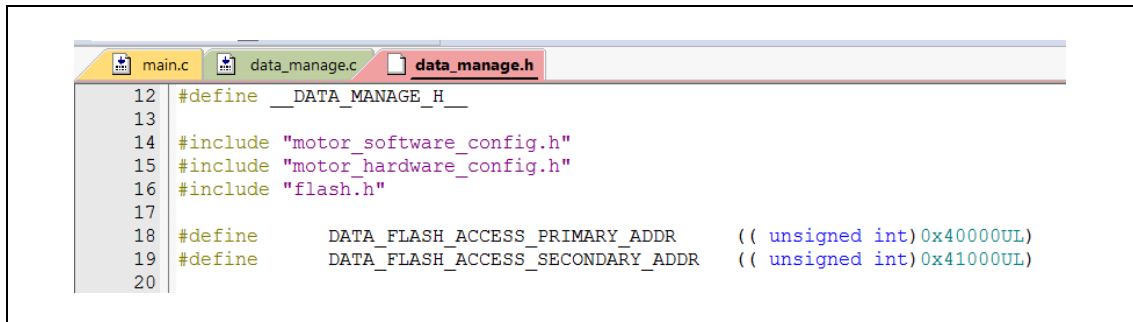


2 调用 API 进行存储

2.1 Data Manage 存储功能

打开 data_manage.h, 先设定好 Sector 的位置, 请注意起始位置必须与 Sector 位置对齐, 地址结尾必须是 00, 不能有数值, 另外也必须注意此段位置不能挪作他用。

图 2-1. 设定存储位置



存储的使用范例如下图, 其中 ai16myData 可以是任一种数据结构之指针(pointer), 而注意此结构大小不得超过 256 byte。

```
/* Put the input value in gsVSP.i16Input */
gsVSP.i16Input = i16VSP;
i16VSPFiltered = AvgFilter_ReadFilterValue(&gsVSP);
ai16myData[0] = i16VSP;
ai16myData[1] = i16VSPFiltered;
```

先将 ADC 读到的数值(i16VSP)存入先定义好的 array 中, 再呼叫以下函数, 将数据存入预先划定好的 sector 内。Page 指标自动+1, 超过 15 自动折回到 1, 均由内部代码自行处理, 使用者即可使用。

```
printf("Data stored...\n\n");
DATA_ByteSeriesSave(DATA_FLASH_ACCESS_PRIMARY_ADDR, /* Sector start address */
                    (uint8_t*)ai16myData,           /* pointer to array stored */
                    sizeof(ai16myData));             /* number of bytes of array */
```

2.2 Data Manage 读取功能

读取功能, 参考代码如下

```
int16_t ai16myData[2];
DATA_ByteSeriesRead(DATA_FLASH_ACCESS_PRIMARY_ADDR, /* Sector start address */
                    (uint8_t*)ai16myData,           /* pointer to array stored */
                    sizeof(ai16myData));             /* number of bytes of array */
printf("Data sread...\n\n");
printf("Task C4 = %d sec\n", u16VTimerC[2]);
printf("VSP = %dmV, ", ai16myData[0]);
printf("Filtered VSP = %dmV\n", ai16myData[1]);
```

只要呼叫此程序, 代码会由 Page0 的纪录自动判断上一次存储时的 Page 指标, 并从中读取最新

的存储信息。

3 修订记录

表 3-1. 文档修订记录

日期	版本	修改内容
2017-10-02	1	初始版本