AN-1452
应用笔记

One Technology Way • P.O. Box 9106 • Norwood, MA 02062-9106, U.S.A. • Tel: 781.329.4700 • Fax: 781.461.3113 • www.analog.com/cn

ADuCM3027/ADuCM3029 Flash EEPROM仿真

简介

非易失性数据存储器是许多嵌入式系统的必备元件。诸如自举配置、校准常数和网络相关信息之类的数据，一般存储在电子可擦除可编程只读存储器（EEPROM）上。采用EEPROM存储这种数据的好处是可以重写或更新EEPROM器件上的单个字节，而不会影响其他位置中的内容。

ADuCM3027/ADuCM3029是集成闪存的超低功耗微控制器单元（MCU）。若在集成闪存上仿真EEPROM，则设计中可省去EEPROM，从而降低BOM成本。同时，软件复杂性也会降低。

背景

闪存通常由页阵列组成。ADuCM3027中的一页为2 kB。写入数据之前必须擦除页面内容。擦除操作适用于整个页面，而读或写操作可针对单一可寻址位置（字节或字）执行。

对单一可寻址位置执行读或写操作有如下挑战：

- 字节宽数据的读和写操作。
- 应能擦除或更新任意位置的数据，同时其他位置的数据保持不变，因为闪存擦除是对整页进行的。

本应用笔记说明利用ADuCM3027/ADuCM3029器件和内置闪存仿真EEPROM的软件，如图1所示。

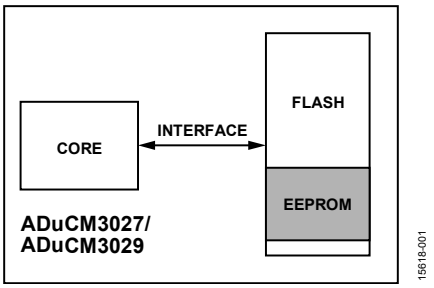

图1. ADuCM3027/ADuCM3029内置闪存和EEPROM系统概览

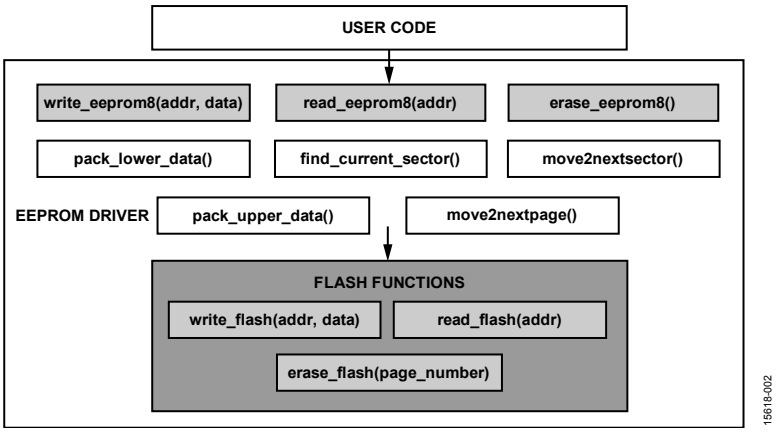

图2. ADuCM3027/ADuCM3029 Flash EEPROM仿真软件结构

目录

简介 1
背景 1
修订历史 2
工作原理 3
EEPROM 3
闪存 5
限制 6
结语 7

修订历史

2017年3月—修订版0：初始版

工作原理

EEPROM仿真需要闪存的一个专用部分。多数EEPROM可以在一个写命令中更新一个字节。然而，只要在两次写操作之间执行擦除序列，闪存器件便能写入多个字节并相应地更新数据。为了在闪存中仿真可读写字节的EEPROM，必须执行读取、更改、写入序列，这与EEPROM操作相似。

本部分介绍的程序使用两个闪存页面（可扩展到两个以上的页面），然后将其划分为多个扇区并用标签加以识别。这种扇区标签提供关于当前正在处理的扇区的信息，以及写入该扇区的数据字节数。注意，每个扇区的最后一个位置保留用于扇区标签，其大小与闪存数据总线的大小相同。扇区大小和一个闪存页面中的扇区数取决于仿真的EEPROM大小。

EEPROM

EEPROM写入和读取函数涉及到应用程序代码输入的处理，如EEPROM数据和地址信息等。EEPROM应用程序编程接口（API）负责根据闪存接口的要求处理和提供数据与地址信息。

写入EEPROM

图3显示了EEPROM写操作流程图。EEPROM写操作的步骤如下：

1. 利用find_current_sector()函数调用找到当前扇区。此搜索基于扇区标签和对应的扇区标签值；返回值为当前扇区起始地址（其为闪存中的一个物理位置）。
2. 借助当前扇区起始地址将EEPROM地址转换为闪存地址。由于ADuCM3027/ADuCM3029闪存具有64位宽数据总线，仿真EEPROM具有8位数据总线，所以软件利用EEPROM地址确定所需的移位次数。
3. 在所获得的闪存地址处读取数据；如果此数据等于0xFF，则屏蔽这些位以创建最低有效位（LSB）和最高有效位（MSB）32位数据包，并对EEPROM数据执行左移位操作以形成待写入闪存的64位宽数据集。
4. 调用write_flash()函数，对闪存控制器执行写命令。此函数的输入参数为闪存地址及LSB和MSB数据包。
5. 对闪存成功执行写操作之后，调用update_tag()函数以更新当前扇区的扇区标签。

如果所获得的闪存地址中已存在数据，则数据读取函数不返回0xFF。这种情况下，通过调用move2nextsector()函数将所获得闪存地址之前或之后的数据移入下一或相邻扇区。将已转换为LSB和MSB数据包的EEPROM数据写入下一扇区上的新闪存地址。这样，每次对EEPROM的一个已写位置执行写操作时，该数据就被移动到下一扇区，其位置中包含修改后的数据。

如果新扇区位于下一页，则调用erase_flash（page_number）函数执行闪存页面擦除命令，在数据移动之后擦除前一页。所有地址寄存器均通过move2nextpage()函数更新。

关于write_eeprom (uint16_t addr_eeprom, uint8_t data_eeprom)函数的详细信息，请参阅表1。

读取EEPROM

图4显示了EEPROM读操作流程图。EEPROM读操作的步骤如下：

1. 调用read_eerprom（addr）函数以读取地址位置处存储的EEPROM值。
2. 在应用程序代码发出的EEPROM读请求中，软件首先确定当前扇区，其中包含最新数据。利用EEPROM地址和当前扇区起始地址获得闪存地址。
3. 利用所获得的闪存地址调用read_flash()函数以执行读命令。
4. 处理从闪存地址接收到的64位宽数据；然后屏蔽、右移此地址的各位，并将其提供给应用程序代码。

关于read_eeprom（uint16_t addr_eeprom）函数的详细信息，请参阅表2。

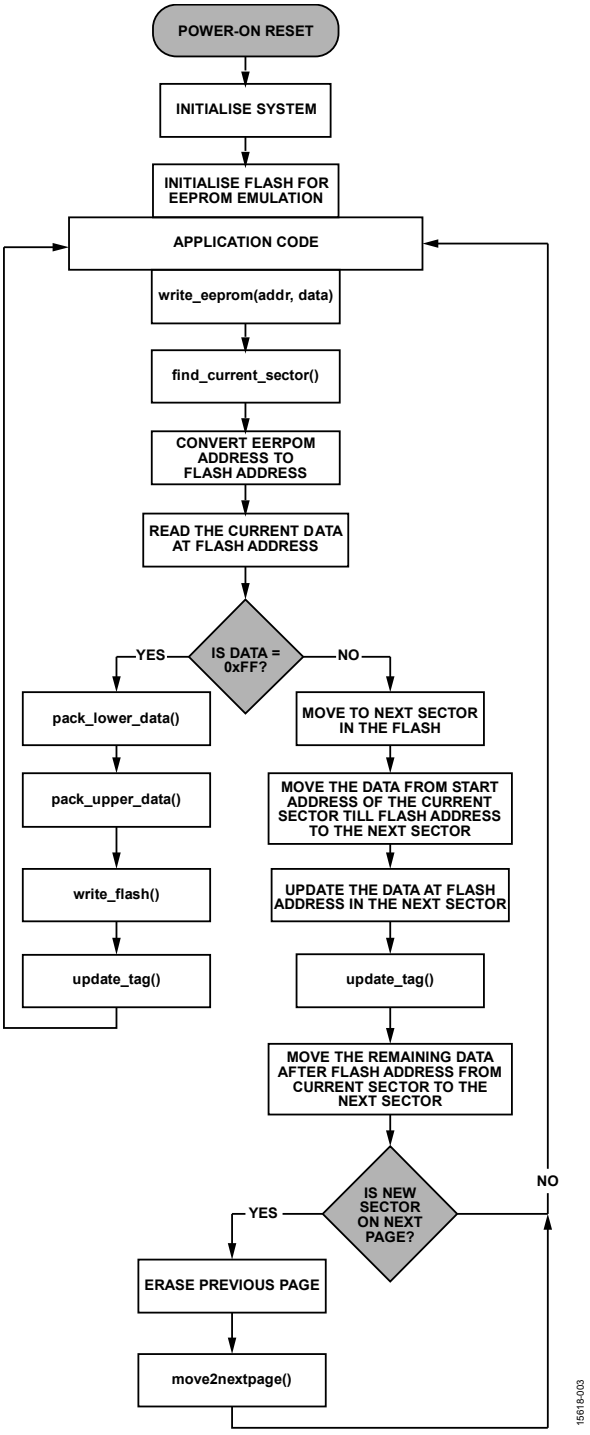

图3. EEPROM写操作

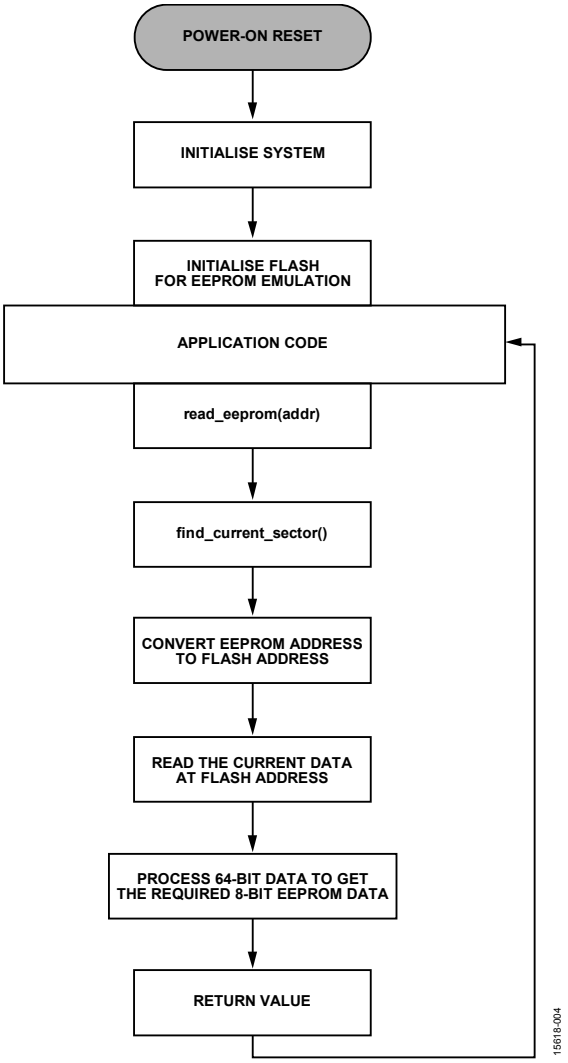

图4. EEPROM读操作

擦除EEPROM

图5显示了EEPROM擦除操作流程图。EEPROM擦除操作的步骤如下：

1. 调用erase_eeprom() 函数以擦除闪存中分配的整个EEPROM空间。

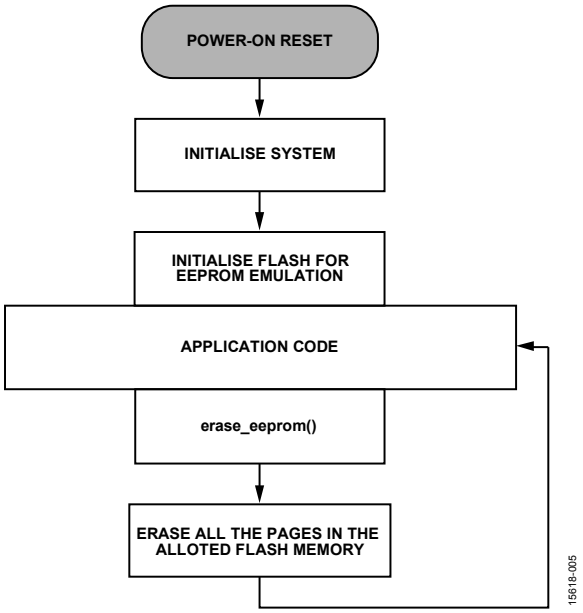

图5. EEPROM擦除操作

闪存中专门用于EEPROM仿真的所有页面均被擦除。因此，在应用程序代码中使用此操作时务必小心。

关于erase_eeprom() 函数的详细信息，请参阅表3。

闪存

ADuCM3027/ADuCM3029处理器包含128 kB和256 kB嵌入式闪存，可通过闪存控制器加以访问。嵌入式闪存具有72位宽数据总线，每次访问提供两个32位字的数据和一个对应的8位纠错码（ECC）字节。闪存组织成页面形式，每页2 kB，另有256字节保留用于ECC。

闪存写入

擦除闪存时，各位置1；写入数据（编程）时，选择性地将相关位清0。没有写操作能将任何位从0置1。因此，一般写访问必须前置一个擦除操作。

锁眼写操作是一种间接写操作，其中用户代码将目标地址和数据值写入存储器映射寄存器，然后命令闪存控制器在后台执行写操作。闪存控制器仅支持通过锁眼写操作对闪存进行写访问。对写访问的这种约束使得闪存控制器能够保证写操作作为原子式双字（64位）操作正确执行。

利用EEPROM数据创建的LSB和MSB数据包提供给锁眼数据寄存器。写命令置位后，闪存控制器启动对给定闪存地址的64位双字写操作。

注意：不支持字（32位）、半字（16位）和字节（8位）写操作。

关于write_flash (uint32_t addr, uint32_t lower_data, uint32_t upper_data) 函数的详细信息，请参阅表4。

闪存读取

只有经过原子式初始化过程之后，才能读取闪存。读取闪存的结果是返回64位双字。

闪存地址信息提供给闪存控制器，后者返回读取的数据。此数据由EEPROM接口进一步处理以实现EEPROM值。

关于read_flash (uint32_t addr) 函数的详细信息，请参阅表6。

闪存擦除

若在写入EEPROM期间有页面变更，则调用erase_flash (page) 函数以对前一页执行页面擦除命令。擦除页面之前会发生数据移动，如上文所述。

关于erase_flash (uint8_t PAGE) 函数的详细信息，请参阅表5。

限制

在实际EEPROM中，若更新一个位置，只会计数一个擦除周期，然后对该特定地址执行写操作，而其他位置保持不变。

在这个仿真EEPROM中，更新一个位置会引起数据从当前扇区移动到下一扇区，这会消耗数目为EEPROM大小的写周期。因此，每次更新一个位置时，数据便移动到下一扇区；如果该扇区位于下一页，则会发生页面擦除操作。这种行为会减少闪存的有效使用期限。

为了克服这些限制，

- 选择用于仿真的EEPROM大小时应小心。小EEPROM可减少数据移动期间的写周期数，间接增加闪存的使用期限。
- 避免不必要的EEPROM写操作。这样可以增加闪存的有效使用期限。例如，系统仅在电源故障序列期间才需要执行写操作。正常工作期间可利用RAM缓冲器存储数据。注意，软件可处理一些对仿真EEPROM的不必要的写操作。例如，若要写入的数据为0xFF，且该特定位置的当前数据为0xFF，则不对闪存执行写操作。

结语

本应用笔记旨在利用ADuCM3027/ADuCM3029比对EEPROM和闪存的差异。仿真EEPROM与实际EEPROM相似，并且消除了与硅面积、输入/输出总线资源、制造成本等相关的问题。

本应用笔记为用户提供了一个较大仿真EEPROM（64字节至1024字节）。仿真EEPROM的大小与闪存使用期限是一对矛盾，建议用户选择适当的大小以增强硬件效率。除了这种方法之外，软件也能处理一些对ADuCM3027/ADuCM3029闪存的不必要的写操作，从而有效增加使用期限。

表1. 写入EEPROM函数描述

write_eeprom (uint16_t addr_eeprom, uint8_t data_eeprom)[1]		
参数	**描述**	**返回值**
addr_eeprom data_eeprom	EEPROM空间中的逻辑地址，数据写入其中 要写入EEPROM空间的数据，addr_eeprom指向该数据	无错误；写操作已成功执行 错误；给定地址超出可用EEPROM存储空间

[1] 此函数用于将数据写入EEPROM。

表2. 读取EEPROM函数描述

read_eeprom (uint16_t addr_eeprom)[1]		
参数	**描述**	**返回值**
addr_eeprom	EEPROM空间中的逻辑地址，数据从其中读取	值；返回8位数据给应用程序代码 错误；给定地址超出可用EEPROM存储空间

[1] 此函数用于从EEPROM读取数据。

表3. 擦除EEPROM函数描述

erase_eeprom ()[1]	
参数	**返回值**
不适用	无错误；擦除操作已成功执行 错误；闪存控制器繁忙，无法执行擦除操作

[1] 此函数用于擦除EEPROM存储空间。注意：若调用此函数，所有数据都会丢失。

表4. 写入闪存函数描述

write_flash (uint32_t addr, uint32_t lower_data, uint32_t upper_data)[1]		
参数	**描述**	**返回值**
addr lower_data upper_data	闪存空间中分配用于EEPROM仿真的地址 双字的低32位 双字的高32位	无错误；写操作已成功执行 错误；给定地址超出可用EEPROM存储空间

[1] 此函数接收来自write_eeprom() 函数的经转换的EEPROM地址和数据，并向闪存控制器发出写命令。

表5. 擦除闪存函数描述

erase_flash (uint8_t PAGE)[1]	
参数	**返回值**
页；分配的闪存空间的页号	无错误；整页擦除已成功执行 错误；给定页值超出分配的闪存空间

[1] 此函数用于在分配的闪存空间中执行整页擦除。

表6. 读取闪存函数描述

read_flash (uint32_t addr)[1]		
参数	**描述**	**返回值**
addr	闪存空间中分配用于EEPROM仿真的地址	读取数据；返回64位数据进行屏蔽，并将其作为返回值发送给read_eeprom() 函数 错误；转换的地址超出分配的闪存空间

[1] 此函数接收来自read_eeprom() 函数的经转换的EEPROM地址，并向闪存控制器发出读命令。

AN15618sc-0-3/17(0)