TB3222

EEPROM Emulation for Flash-Only Devices

Introduction

When migrating from 8-bit MCUs to 32-bit MCUs, one of the biggest obstacles is the lack of real EEPROM memory in majority of 32-bit MCUs. Several software emulation layers exist to address this issue. However, while providing a seamless emulation of the EEPROM interface, they all have significant drawbacks like excessive wear on the underlying Flash memory and may result in complete data loss in case of power interruption while the write is ongoing.

A better approach is described in this document. This approach requires minor changes to the application flow, but yields a robust EEPROM emulation with minimal wear of the MCU Flash memory.

Table of Contents

1. The Problem

Given the lack of an EEPROM in the majority of modern 32-bit MCUs, the only available location for storing the variable data is the Flash memory.

The disadvantage of the Flash memory is that it cannot be erased or written in single bytes, as is typically done for EEPROM memory. Flash memory can only be erased and written in large blocks. A typical erase block size may be 256 to 8192 bytes, and a typical write block is 64 to 512 bytes. Some Flash implementations support partial writes, a mode where an erased block may be written multiple times as long as bits only changed from “1” (erased state) to “0” (programmed state). Even if the Flash supports partial writes, it often comes with further limitations. In some cases, there is a limit on the number of partial writes that may be performed before an erase is necessary. In other cases, writes must happen in some specific alignment and size combinations.

Additionally, the Flash memory allocated for the data storage is often located in the same Flash array as the application code. This leads to the need to pause the application execution while data is written. The implementation proposed here enables the user to define the point at which this block is going to happen. On the other hand, MCU manufacturers recognize this problem, and in many modern MCUs there is a separate data Flash section available for this use. In some cases, the main Flash array is split into multiple independent banks, which can be written while other banks are available for reading and code execution.

2. The Solution

The proposed solution is to combine all the variables to be stored persistently in a structure along with a few service members. The structure can be mapped to the address in the Flash memory or copied into the SRAM for faster access and possibility of cumulative updates.

The following code shows a typical structure for the data layout. The first part shows the original structure, and the second part shows the layout after two version updates in addition to the original version.

```
---------------------------------------
typedef struct
{
  uint32_t     counter;
  uint8_t      version;
  // User Values
  uint32_t     crc;
} EepromLayout_v1;

typedef struct
{
  uint32_t     counter;
  uint8_t      version;
  // Version 1 User Values
  uint32_t     crc1;
  // Version 2 User Values
  uint32_t     crc2;
  // Version 3 User Values
  uint32_t     crc3;
} EepromLayout_v3;
---------------------------------------
```

The version field identifies the version of the structure. Applications that do not have firmware updates and implement a fixed layout, may omit the version field.

The counter field is a monotonically incrementing counter. Along with CRC field, the highest counter value identifies the recent valid version of the structure. The 32-bit value will never overflow in practice, because the counter is only incremented on writes, and the Flash write endurance is limited.

The size of the structure for CRC calculation may be inferred from the version field, or explicitly defined at part of the structure.

The robustness of this approach comes from allocating the storage for multiple copies of the structure. Multiple copies may be allocated in the same erase block if the Flash supports partial writes, but the care must be taken to allocate at least two erase blocks. At the same time at least two copies must be allocated. This will ensure data protection against damage in case of power interruptions when one of the blocks must be erased.

Allocating multiple copies has an additional benefit of implementing wear leveling at no additional cost.

The size of the data allocated for each copy must be enough to cover the current size of the structure. It must also be big enough to cover possible future extensions with the new versions. Updating the allocation size with the firmware update to a new version is possible but may be complicated.

At the time of the initialization, the application must iterate over all possible copies of the structure. On each iteration it must calculate the CRC of the current copy and compare the counter field to the best candidate so far. The copy with the higher counter and version values and the valid CRC value is the recent valid data that must be loaded into SRAM and used by the application.

If no valid copy is found, the application must initialize all fields to their default value. This can only happen on a freshly programmed device.

The application may detect that the copy is valid, but the version of the structure is lower than currently supported by the application. This will happen on the first boot after the firmware update that changes the layout of the data. In this case parts of the data from the old version should be loaded as is, and the new fields should be initialized to the default values. The version field should be updated to the latest value, after that the structure will be saved in the new version format the next time it is written.

Applications that support firmware updates must support all prior versions of the structure in order to load the data in the old format. If a firmware downgrade is supported, then the application must preserve the layout of the data structure to be compatible with the old versions. This includes keeping and updating the CRC values for all previous versions. If firmware downgrade is not possible, then only one CRC may be used for the recent version of the structure.

If firmware downgrade is supported, then the version field indicates the latest version. After firmware downgrade, the application may see an unsupported version set in the version field. In this case it must assume that the version was set by the recent firmware and use the CRC alone as an indicator for the validity of the data.

When saving the data to the Flash, the index of the next entry to be written is calculated as (C + 1) % N, where C is the index of the current valid entry and N is the total number of entries allocated for the persistent storage.

After a structure is saved, if the new index is pointing to another erase block, the application may choose to erase the next block ahead of time. This will move erase operation to the time of the last write, making the next write much faster.

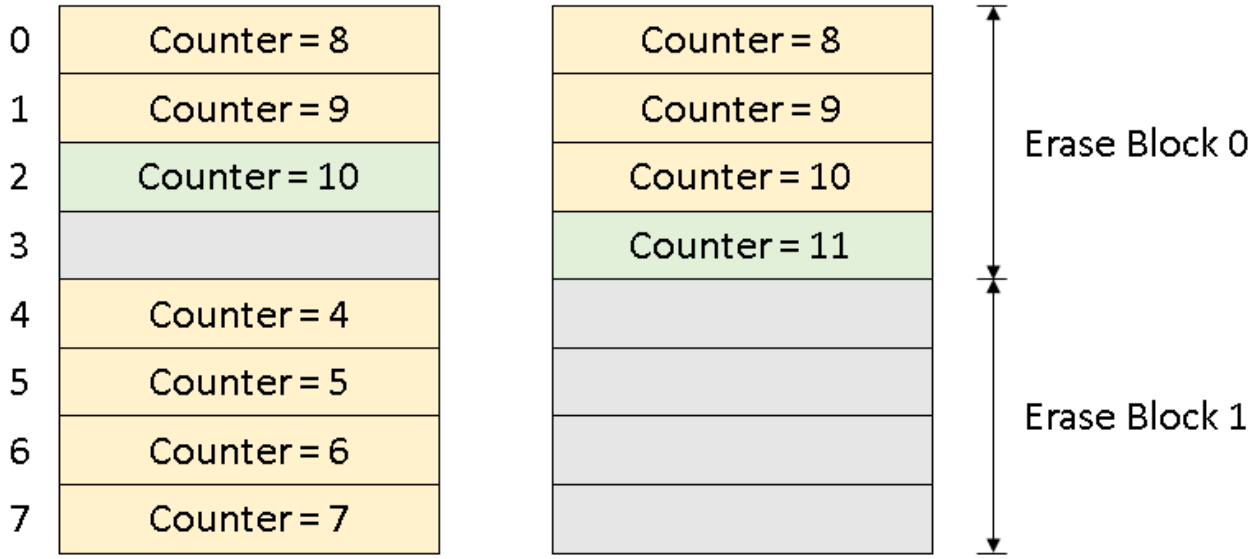
Typically write operation is much faster than erase. Erasing the block ahead of time may give enough time to the application to detect power failure and save the currently unsaved data.

After the initialization, the application cannot assume that the next block was properly erased after the previous write, since it may have been interrupted by the power failure. Application must always perform an erase in this case, even if the Flash appears to be erased.

3. Practical Implementation Considerations

The following figure shows a typical memory layout. In the memory layout, eight copies are allocated, that is, four copies for each erase block. In the first case the entry with the index 2 (Counter = 10) is the recent copy. After the data is updated, the entry with the index 3 (Counter = 11) will be the recent. Simultaneously the erase block ‘1’ is erased, because it will contain the next written entry. All entries are used in a round-robin fashion, ensuring equal wear of the Flash memory.

Figure 3-1. Memory Layout



To limit the number of Flash erase cycles, an actual write must be initiated by the application. This may happen on each field update, or after several fields were updated simultaneously. Additionally, the application may choose to save updated values periodically after a set timeout. Any changes to the data within the same timeout period will be written simultaneously saving Flash erase cycles.

In cases, when EEPROM access is rare and mostly read-only, for example, when EEPROM is used to store calibration data, it may make more sense to access the data directly from the Flash without maintaining a copy in the SRAM. In this case a temporary copy of the data structure must be created in the SRAM every time the data must be saved.

Special care must be taken when the persistent memory is allocated in the separate data section of the device that is accessible for writes while the main Flash is available for the code execution. In this case the code is executed in parallel with the Flash write and erase operations, which minimized the impact of the EEPROM emulation on the code execution. However, the EEPROM emulation code must keep track of the fields that are changed during the ongoing write. The data saving code may either create a copy of the data that is being saved, or block execution of the update operations while Flash access is in progress.

The Microchip Website

Microchip provides online support via our website at http://www.microchip.com/. This website is used to make files and information easily available to customers. Some of the content available includes:

- **Product Support** – Data sheets and errata, application notes and sample programs, design resources, user's guides and hardware support documents, latest software releases and archived software
- **General Technical Support** – Frequently Asked Questions (FAQs), technical support requests, online discussion groups, Microchip design partner program member listing
- **Business of Microchip** – Product selector and ordering guides, latest Microchip press releases, listing of seminars and events, listings of Microchip sales offices, distributors and factory representatives

Product Change Notification Service

Microchip's product change notification service helps keep customers current on Microchip products. Subscribers will receive email notification whenever there are changes, updates, revisions or errata related to a specified product family or development tool of interest.

To register, go to http://www.microchip.com/pcn and follow the registration instructions.

Customer Support

Users of Microchip products can receive assistance through several channels:

- Distributor or Representative
- Local Sales Office
- Embedded Solutions Engineer (ESE)
- Technical Support

Customers should contact their distributor, representative or ESE for support. Local sales offices are also available to help customers. A listing of sales offices and locations is included in this document.

Technical support is available through the web site at: http://www.microchip.com/support

Microchip Devices Code Protection Feature

Note the following details of the code protection feature on Microchip devices:

- Microchip products meet the specification contained in their particular Microchip Data Sheet.
- Microchip believes that its family of products is one of the most secure families of its kind on the market today, when used in the intended manner and under normal conditions.
- There are dishonest and possibly illegal methods used to breach the code protection feature. All of these methods, to our knowledge, require using the Microchip products in a manner outside the operating specifications contained in Microchip's Data Sheets. Most likely, the person doing so is engaged in theft of intellectual property.
- Microchip is willing to work with the customer who is concerned about the integrity of their code.
- Neither Microchip nor any other semiconductor manufacturer can guarantee the security of their code. Code protection does not mean that we are guaranteeing the product as "unbreakable."

Code protection is constantly evolving. We at Microchip are committed to continuously improving the code protection features of our products. Attempts to break Microchip's code protection feature may be a violation of the Digital Millennium Copyright Act. If such acts allow unauthorized access to your software or other copyrighted work, you may have a right to sue for relief under that Act.

Legal Notice

Trademarks

ISBN: 978-1-5224-4756-6

Quality Management System

For information regarding Microchip's Quality Management Systems, please visit http://www.microchip.com/quality.

Worldwide Sales and Service

AMERICAS	ASIA/PACIFIC	ASIA/PACIFIC	EUROPE
Corporate Office 2355 West Chandler Blvd. Chandler, AZ 85224-6199 Tel: 480-792-7200 Fax: 480-792-7277 Technical Support: http://www.microchip.com/support Web Address: http://www.microchip.com **Atlanta** Duluth, GA Tel: 678-957-9614 Fax: 678-957-1455 **Austin, TX** Tel: 512-257-3370 **Boston** Westborough, MA Tel: 774-760-0087 Fax: 774-760-0088 **Chicago** Itasca, IL Tel: 630-285-0071 Fax: 630-285-0075 **Dallas** Addison, TX Tel: 972-818-7423 Fax: 972-818-2924 **Detroit** Novi, MI Tel: 248-848-4000 **Houston, TX** Tel: 281-894-5983 **Indianapolis** Noblesville, IN Tel: 317-773-8323 Fax: 317-773-5453 Tel: 317-536-2380 **Los Angeles** Mission Viejo, CA Tel: 949-462-9523 Fax: 949-462-9608 Tel: 951-273-7800 **Raleigh, NC** Tel: 919-844-7510 **New York, NY** Tel: 631-435-6000 **San Jose, CA** Tel: 408-735-9110 Tel: 408-436-4270 **Canada - Toronto** Tel: 905-695-1980 Fax: 905-695-2078	**Australia - Sydney** Tel: 61-2-9868-6733 **China - Beijing** Tel: 86-10-8569-7000 **China - Chengdu** Tel: 86-28-8665-5511 **China - Chongqing** Tel: 86-23-8980-9588 **China - Dongguan** Tel: 86-769-8702-9880 **China - Guangzhou** Tel: 86-20-8755-8029 **China - Hangzhou** Tel: 86-571-8792-8115 **China - Hong Kong SAR** Tel: 852-2943-5100 **China - Nanjing** Tel: 86-25-8473-2460 **China - Qingdao** Tel: 86-532-8502-7355 **China - Shanghai** Tel: 86-21-3326-8000 **China - Shenyang** Tel: 86-24-2334-2829 **China - Shenzhen** Tel: 86-755-8864-2200 **China - Suzhou** Tel: 86-186-6233-1526 **China - Wuhan** Tel: 86-27-5980-5300 **China - Xian** Tel: 86-29-8833-7252 **China - Xiamen** Tel: 86-592-2388138 **China - Zhuhai** Tel: 86-756-3210040	**India - Bangalore** Tel: 91-80-3090-4444 **India - New Delhi** Tel: 91-11-4160-8631 **India - Pune** Tel: 91-20-4121-0141 **Japan - Osaka** Tel: 81-6-6152-7160 **Japan - Tokyo** Tel: 81-3-6880- 3770 **Korea - Daegu** Tel: 82-53-744-4301 **Korea - Seoul** Tel: 82-2-554-7200 **Malaysia - Kuala Lumpur** Tel: 60-3-7651-7906 **Malaysia - Penang** Tel: 60-4-227-8870 **Philippines - Manila** Tel: 63-2-634-9065 **Singapore** Tel: 65-6334-8870 **Taiwan - Hsin Chu** Tel: 886-3-577-8366 **Taiwan - Kaohsiung** Tel: 886-7-213-7830 **Taiwan - Taipei** Tel: 886-2-2508-8600 **Thailand - Bangkok** Tel: 66-2-694-1351 **Vietnam - Ho Chi Minh** Tel: 84-28-5448-2100	**Austria - Wels** Tel: 43-7242-2244-39 Fax: 43-7242-2244-393 **Denmark - Copenhagen** Tel: 45-4450-2828 Fax: 45-4485-2829 **Finland - Espoo** Tel: 358-9-4520-820 **France - Paris** Tel: 33-1-69-53-63-20 Fax: 33-1-69-30-90-79 **Germany - Garching** Tel: 49-8931-9700 **Germany - Haan** Tel: 49-2129-3766400 **Germany - Heilbronn** Tel: 49-7131-72400 **Germany - Karlsruhe** Tel: 49-721-625370 **Germany - Munich** Tel: 49-89-627-144-0 Fax: 49-89-627-144-44 **Germany - Rosenheim** Tel: 49-8031-354-560 **Israel - Ra'anana** Tel: 972-9-744-7705 **Italy - Milan** Tel: 39-0331-742611 Fax: 39-0331-466781 **Italy - Padova** Tel: 39-049-7625286 **Netherlands - Drunen** Tel: 31-416-690399 Fax: 31-416-690340 **Norway - Trondheim** Tel: 47-72884388 **Poland - Warsaw** Tel: 48-22-3325737 **Romania - Bucharest** Tel: 40-21-407-87-50 **Spain - Madrid** Tel: 34-91-708-08-90 Fax: 34-91-708-08-91 **Sweden - Gothenberg** Tel: 46-31-704-60-40 **Sweden - Stockholm** Tel: 46-8-5090-4654 **UK - Wokingham** Tel: 44-118-921-5800 Fax: 44-118-921-5820