life.augmented

AN4894
アプリケーション･ノート

STM32 マイクロコントローラ向けの EEPROM エミュレーションの手法とソフトウェア

概要

EEPROM（電気的消去･プログラム可能読み取り専用メモリ）は、更新可能なアプリケーション･データの不揮発記憶または複雑なシステムでの電源障害に備えた少量のデータ保持などに使用されます。コスト削減のために、固有のソフトウェア･アルゴリズムの使用を前提に、外付けの EEPROM を内蔵 Flash メモリに置き換えることが可能です。

本アプリケーション･ノートでは、スタンドアロンの EEPROM を置換するソフトウェア･ソリューション（X-CUBE-EEPROM）について説明します。このソフトウェアは、表 1：対象製品に示した STM32 シリーズに搭載された内蔵 Flash メモリを使用して EEPROM のメカニズムをエミュレートします。X-CUBE-EEPROM では、この EEPROM エミュレーション･ドライバの活用方法を示す例を含むファームウェア･パッケージも提供しています（セクション 5：API およびアプリケーション例参照）。

STM32WB シリーズ製品に限っては、EEPROM 動作の実行中に Bluetooth® Low Energy による接続と通信を維持する例も用意しました。

このエミュレーション手法は、Flash メモリ･ページを少なくとも 2 ページ使用し、EEPROM エミュレーションのコードは、ページがフルになると両ページ間でデータをスワップします。この動作はユーザからは見えません。このアプリケーション･ノートとともに提供される EEPROM エミュレーション･ドライバには、次のような特徴があります。

- 軽量な実装と縮小されたフットプリント
- Flash メモリのフォーマット、初期化、データの読み書き、ページのクリーンアップに必要な少数の関数からなるシンプルな API
- 内部データ管理のために、Flash メモリ･ページを少なくとも 2 ページ使用
- ユーザのクリーンアップ操作を簡素化（バックグラウンドのページ消去）
- エミュレート EEPRM の E/W サイクル耐性を向上させるウェアレベリング･アルゴリズム
- 非同期のリセットや電源障害に対する堅牢性
- マルチコア STM32 デバイス（STM32WB シリーズなど）においてコア間で共有される Flash メモリを保護する機能の実装（任意）
- キャッシュ･コヒーレンシの維持

エミュレートする EEPROM のサイズは柔軟に選択でき、エミュレート目的に割り当てられる Flash メモリのサイズのみで決まります。

表 1. 対象製品

タイプ	シリーズ
マイクロコントローラ	STM32L4 シリーズ、STM32L4+ シリーズ、STM32L5 シリーズ、STM32G0 シリーズ、STM32G4 シリーズ、STM32WB シリーズ、STM32WL シリーズ

目次

表の一覧

図の一覧

1 一般情報

本ドキュメントの適用範囲は、ARM ®(1) コアに基づく STM32 マイクロコントローラです。

1.1 参照文献

EEPROM エミュレーションのソリューションやアプリケーション·ノートは、下記の参照文献 [1] に示したとおり、他の STM32 シリーズについても提供されています。

表 2. 関連ドキュメント

参照番号	ドキュメント名
[1]	アプリケーション·ノート： – STM32F0 シリーズ：STM32F0xx マイクロコントローラでの EEPROM エミュレーション（AN4061） – STM32F1 シリーズ：STM32F10x マイクロコントローラでの EEPROM エミュレーション（AN2594） – STM32F2 シリーズ：STM32F2xx マイクロコントローラでの EEPROM エミュレーション（AN3390） – STM32F3 シリーズ：STM32F3xx マイクロコントローラでの EEPROM エミュレーション（AN4046）/ STM32F30x/STM32F31x STM32F37x/STM32F38x マイクロコントローラの EEPROM エミュレーション（AN4056） – STM32F4 シリーズ：STM32F40x/STM32F41x マイクロコントローラでの EEPROM エミュレーション（AN3969）
[2]	STM32WB シリーズ·マイクロコントローラでワイヤレス·アプリケーションを構築する、アプリケーション·ノート（AN5289）

1. Arm は、米国内およびその他の地域にある Arm Limited 社（またはその子会社）の登録商標です。

2 外付け EEPROM とエミュレート EEPROM の主な相違

EEPROM は、実行時に、バイト、ハーフワード、ワード単位で更新される不揮発のデータ記憶領域を必要とする、多くの組込みアプリケーションにおいて重要な役割を担う部品です。しかし、こうしたシステムで使用されるマイクロコントローラでは、多くの場合それ自体に内蔵された Flash メモリに基づいて動作しています。部品点数の削減や PCB 面積の縮小によりシステム·コストを抑えるために、外付け EEPROM の代わりに STM32 の Flash メモリを使用して、コードだけでなくデータも保存することが可能です。

内蔵 Flash メモリにデータを保存するには、特別なソフトウェア管理が必要になります。EEPROM エミュレーションのソフトウェア方式は、EEPROM に要求される信頼性、使用する Flash メモリのアーキテクチャ、最終製品の要件、その他の各種パラメータなど、多くの要因に依存します。

内蔵 Flash メモリと外付けのシリアル EEPROM の主な相違点は、Flash メモリ·テクノロジーを使用しているマイクロコントローラであればどれも同じであり、STM32 シリーズのマイクロコントローラに固有のものではありません。相違の一つが、EEPROM ではデータの再書込みに先立って、空き領域を確保するための消去動作が不要であるという点です。その他の主な相違点を表 3 にまとめました。

表 3. 外付け EEPROM とエミュレート EEPROM の主な相違

機能	外付け EEPROM （例：M24C64：I²C シリアル·アクセス EEPROM）	内蔵 Flash メモリによるエミュレート EEPROM
書込み時間	ランダム·バイト書込み時間 = 4 ms ワード·プログラム時間 = 16 ms ページ（32 バイト）書込み時間 = 4 ms連続ワード·プログラム時間 = 500 µs	ワード·プログラム時間 = 90 µs ~ 580 ms (1)
消去時間	N/A	2 KB ページ消去時間 = 22 ms など (2)
メモリ·サイズ	数 KB ~ 256 KB	制限は EEPROM エミュレーションに使用できる Flash メモリのサイズだけで決まります。
読出しアクセス	シリアル = 100 µs ランダム·ワード = 92 µs ページ：22.5 µs/バイト	並列：アクセス時間 = 6 µs ~ 592 µs (1)
書換え耐性	4Mサイクル @25 ℃ 1.2Mサイクル @85 ℃ 600Kサイクル @125 ℃	ページあたり 10Kサイクル @105℃(2) 内蔵 Flash メモリの複数のページを使用すると、書込みサイクル数が増加したことと同じ効果が得られます。セクション 4.4 : 書換えサイクル能力 : EEPROM 書換え耐性の向上を参照してください。
データ保持期間 (2)	50 年 @125 ℃ 100 年 @25 ℃	7 年 @125 ℃ 15 年 @105 ℃ 30 年 @85 ℃

1. 詳細については、第 5.3 章 : EEPROM エミュレーションのタイミングを参照してください。
2. STM32L4 シリーズのデータ例。使用する STM32 製品のデータシートをご覧ください。

2.1 書込みアクセス時間の差異

Flash メモリは書込みアクセス時間が短いため、ほとんどの場合、エミュレート EEPROM には、外付け EEPROM よりも高速に重要なパラメータを保存できます。ただし、データ転送のメカニズムにより、エミュレート EEPROM の書込みアクセス時間が、外付け EEPROM よりも大幅に長くなる場合もあります。

2.2 プログラムおよび消去動作

Flash メモリと違って EEPROM ではプログラム済みのアドレスに書き込む際に、あらかじめ空き領域を確保するための消去動作が必要ありません。これは単体の（スタンドアロン）EEPROM と内蔵 Flash メモリによるエミュレート EEPROM の間の大きな違いです。

- 内蔵 Flash メモリによってエミュレートした EEPROM

  消去プロセスの管理は、すべてを EEPROM エミュレーション･ソフトウェアが行いますが、消去動作はアプリケーション･ソフトウェアに任されています。これにより、ワースト･ケースの書込み時間を短縮でき、またアプリケーションの実行時間が重要でなくなった場合に Flash ページ消去動作が可能になります。

  さらに、Flash メモリのプログラムおよび消去動作には極めて長時間を要するため、Flash メモリ管理ソフトウェアの設計時には、消去プロセスを中断させる可能性がある電源障害その他のスプリアス･イベント（リセットなど）が考慮されています。EEPROM エミュレーション･ソフトウェアは、電源障害や完全に非同期のリセットに対して高い堅牢性を持つように設計されています。

- スタンドアロンの外付け EEPROM

  CPU によって開始されたワード書込みは、CPU リセットによっても中断されません。電源障害だけが書込みプロセスを中断させる可能性があるため、スタンドアロン EEPROM 内の書込みプロセスを確実に完了できるように、電源監視と適切な容量のデカップリング･コンデンサが必要になります。

3 EEPROM エミュレーションの実装

3.1 原理

EEPROM エミュレーションは、Flash メモリの特性や最終製品の要件を考慮して、さまざまな方法で実現できます。以下に詳述する手法では、不揮発データに割り当てるために Flash メモリの複数のページを含むページ･セットを 2 つ必要とします

第 1 のページ･セットは、初期状態として消去されており、新規データの保存に使用します。Flash のプログラム動作は Flash アドレスの昇順にシーケンシャルに実行されます。第 1 のページ･セットがデータで満たされたら、ガーベジ･コレクション（領域の解放と再利用）が必要になります。

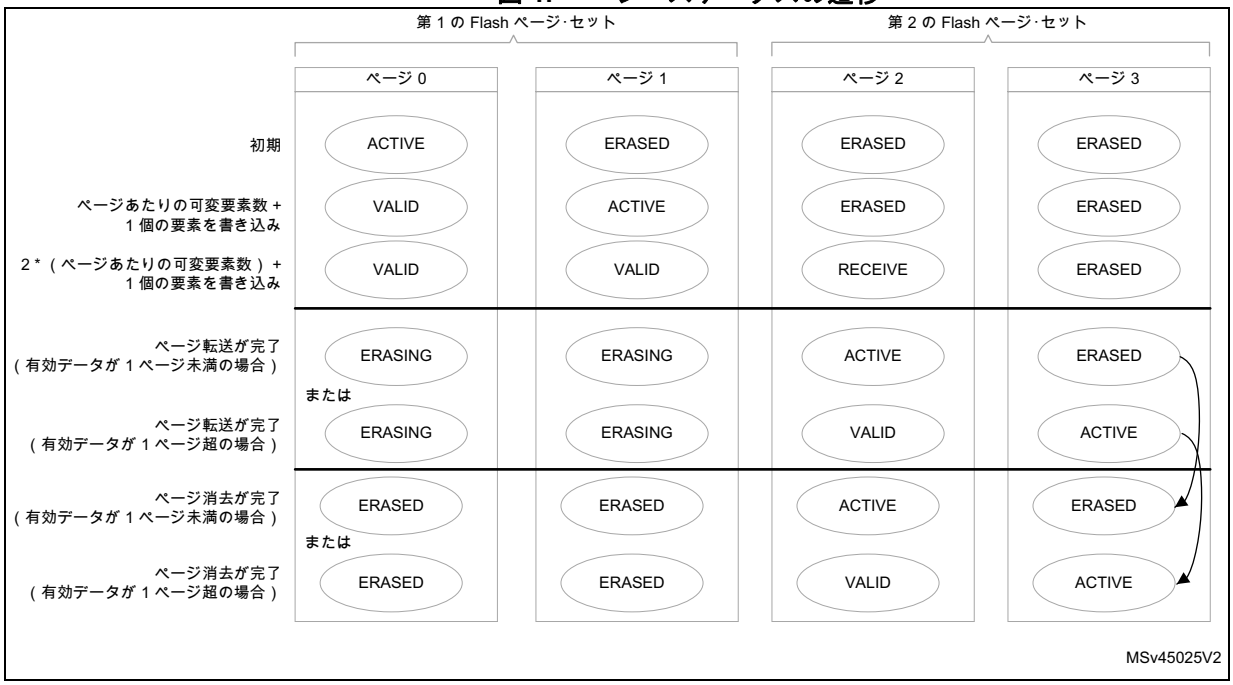
第 2 のページ･セットに第 1 のページ･セットの有効なデータだけを収集し、残りの領域は新規データの保存に使用できます。有効なデータを第 2 のページ･セットに移し終えたら、第 1 のページ･セットは消去可能になります。

各ページ･セットは、1 つまたは複数の Flash ページから構成されます。各ページの先頭 4 つの 64 bit ワード（32 バイト）を占めるヘッダ･フィールドは、各ページの状態を示します。各ページには、次の 5 通りの状態があります。

- **ERASED**（消去済み）: ページがエンプティ（初期状態）です。
- **RECEIVE**（受信）: 他のフルのページからデータを受信するデータ転送のためにページを使用中です。
- **ACTIVE**（アクティブ）: 新規データを保存するためにページを使用中です。
- **VALID**（有効）: ページがフルです。この状態は、有効なすべてのデータが受信（RECEIVE）ページに完全に転送されるまで変化しません。
- **ERASING**（消去中）: このページの有効なデータの転送が完了しています。ページは、いつでも消去できます。

図 1 に各ページ･セットが 2 ページで構成されている場合の、ページ・ステータス遷移を示します。

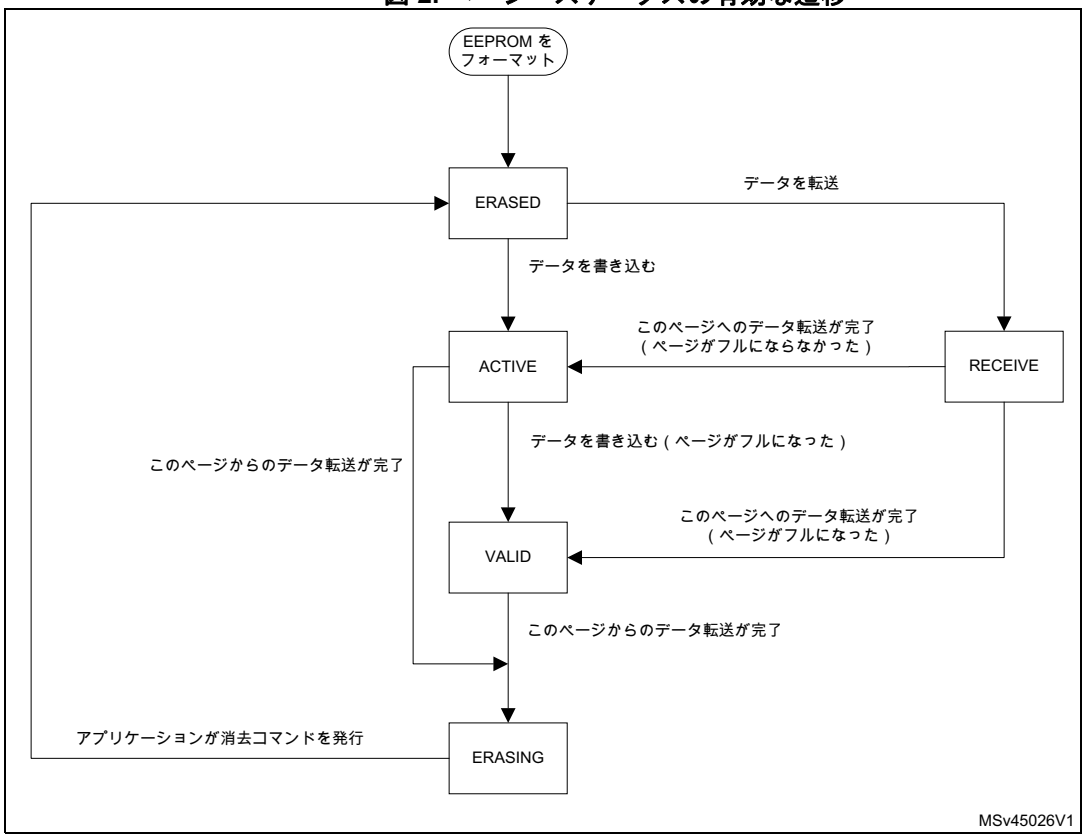
図 1. ページ・ステータスの遷移



3.2 ページ・ステータスの有効な遷移

本節では、EEPROM エミュレーション·ドライバの改造を目指すユーザのみに役立つ情報を提供します。単にドライバを使用するユーザには必要のない情報です。

図 2. ページ・ステータスの有効な遷移



3.3 ページと可変要素のフォーマット

Flash メモリのページ·サイズと構成は STM32 のシリーズによって異なる場合があります。

さらに、一部の STM32 シリーズでは、EEPROM エミュレーション·ドライバが、シングル·バンク·モードまたはデュアル·バンク·モードのいずれか一方、または両方をサポートしています。EEPROM エミュレーション·ドライバを使用する際は、サポートされるモードに応じて適切なモードを選択する必要があります。詳細については、表 4 を参照してください。

表 4. Flash メモリのプロパティ

STM32 シリーズ / バリュー·ライン	Flash メモリ·ページ·サイズ	シングル·バンク·モードへの対応	デュアル·バンク·モードへの対応
STM32L4+ シリーズ	4 KB（DBANK = 1 の場合 64 bit x 512 ワード）	不可(1)	可
STM32L41x/42x/43x 44x/45x/46x	2 KB（64 bit x 256 ワード）	可	不可
STM32L47x/48x/49x/4Ax	2 KB（64 bit x 256 ワード）	可	可
STM32L552：STM32L562	2 KB（DBANK = 1 の場合 64 bit x 256 ワード）	不可 (1)	可
STM32G0x0 STM32G0x1	2 KB（64 bit x 256 ワード）	可	不可
STM32G4 シリーズ	2 KB（DBANK = 1 の場合 64 bit x 256 ワード）	不可 (1)	可
STM32WB シリーズ	4 KB（64 bit x 512 ワード）	可	不可
STM32WL シリーズ	2 KB（256 バイト x 8 行）	可	不可

1. シングル·バンク·モードの場合、これらのデバイスは 128 bit 幅の読出しアクセスを実行します。しかし、EEPROM エミュレーション·ソリューションは、64 bit 幅の読出しアクセス向けに設計されています。

Flash メモリの最小書込み幅は 64 bit です。これはECC（エラー訂正コード）機能がオフにできないためです。既にプログラムされてヌル以外の値を持つ Flash 行にはゼロ（0x0000000000000000）しか書き込むことができません。先頭の 4 ワードはヘッダとして使用されるため、Flash の 1 ページには、ページ·サイズが 2 KB の場合、最大で 252 個の可変要素、ページ·サイズが 4 KB の場合、508 個の可変要素を保存できます。

Flash ページが取りうる状態は、ページ·ヘッダに 0xAAAAAAAAAAAAAAAA を書き込むことでコーディングします。ページの状態は、次の手順によって判断できます。

- 4 行目が消去されていない場合、ページの状態は ERASINGステータスです。
- 3 行目が消去されてなく、4 行目が消去状態の場合、ページの状態は VALID です。
- 2 行目が消去されてなく、3 行目と 4 行目が消去状態の場合、ページの状態は ACTIVE です。
- 1 行目が消去されてなく、2 ~ 4 行目が消去状態の場合、ページの状態は RECEIVE です。
- 先頭 4 行が消去されていない場合、ページの状態は ERASED です。

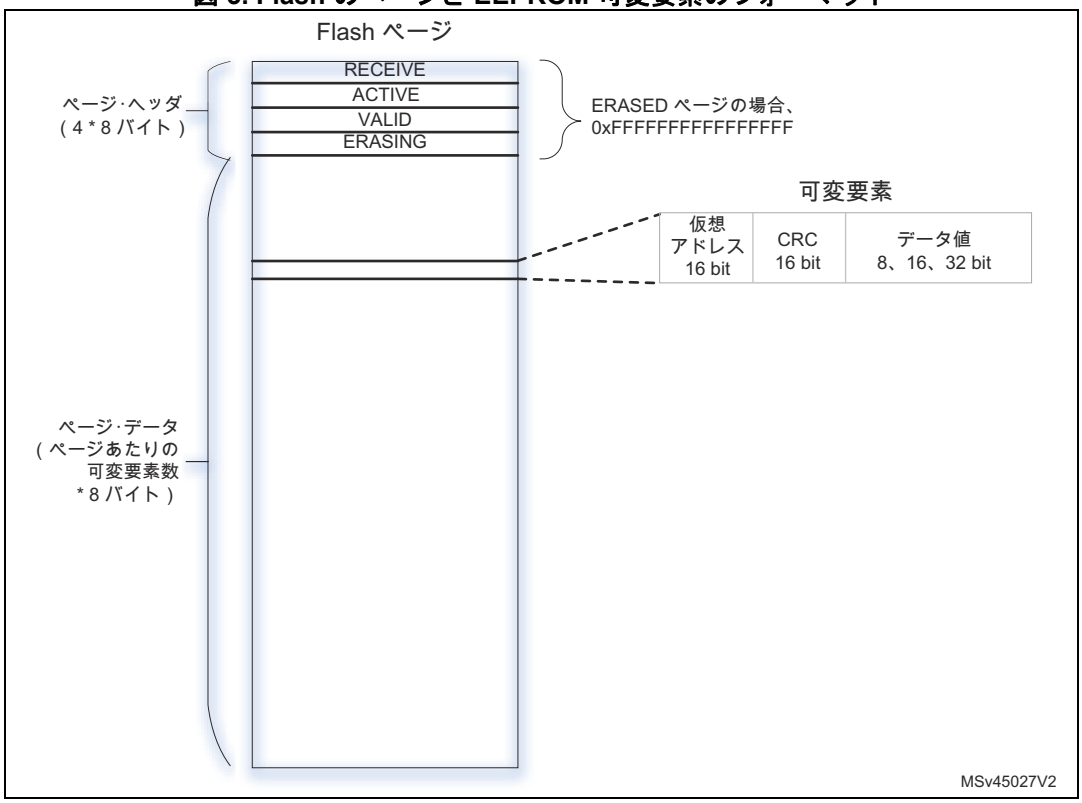
このアルゴリズムにより、セクション 3.2：ページ・ステータスの有効な遷移で説明しているすべての状態と遷移のコーディングが可能になります。

各可変要素（保存したいデータ）は、仮想アドレスとデータ値によって定義され、以降のデータ読出しまたは更新に備えて、Flash メモリに格納されます。実装されるソフトウェアでは、仮想アドレスの長さが 16 bit、データ値の長さは 8、16、32 bit のいずれかです。ドライバでは、0x0001（0x0000 はドライバによって無効化された EEPROM 要素に対応します）から 必要な EEPROM 可変要素の最大数までの仮想アドレスの値が必要になります。また、仮想アドレスが 16 bit 幅であることから、EEPROM 可変要素の最大数は 0xFFFE を超えることができません（0xFFFF は、消去済みの Flash 行に対応します）。さらに、可変要素の数は、製品の Flash メモリのサイズによっても制限されます（セ

クション 4.5 : EEPROM エミュレーションに必要な Flash サイズの計算を参照してください)。

各要素には、その整合性の確認に使用する 16 bit の CRC も含まれます。データを変更した場合、変更済みのデータに、変更前と同じ仮想アドレスを関連付けて、新しい Flash メモリ位置に格納します。データ読出しでは、最新のデータ値を返します。

図 3. Flash のページと EEPROM 可変要素のフォーマット



3.4 簡単な使用例

以下に、表 5 に示す仮想アドレスを持つ 3 つの EEPROM 可変要素をソフトウェアによって管理する例を示します。

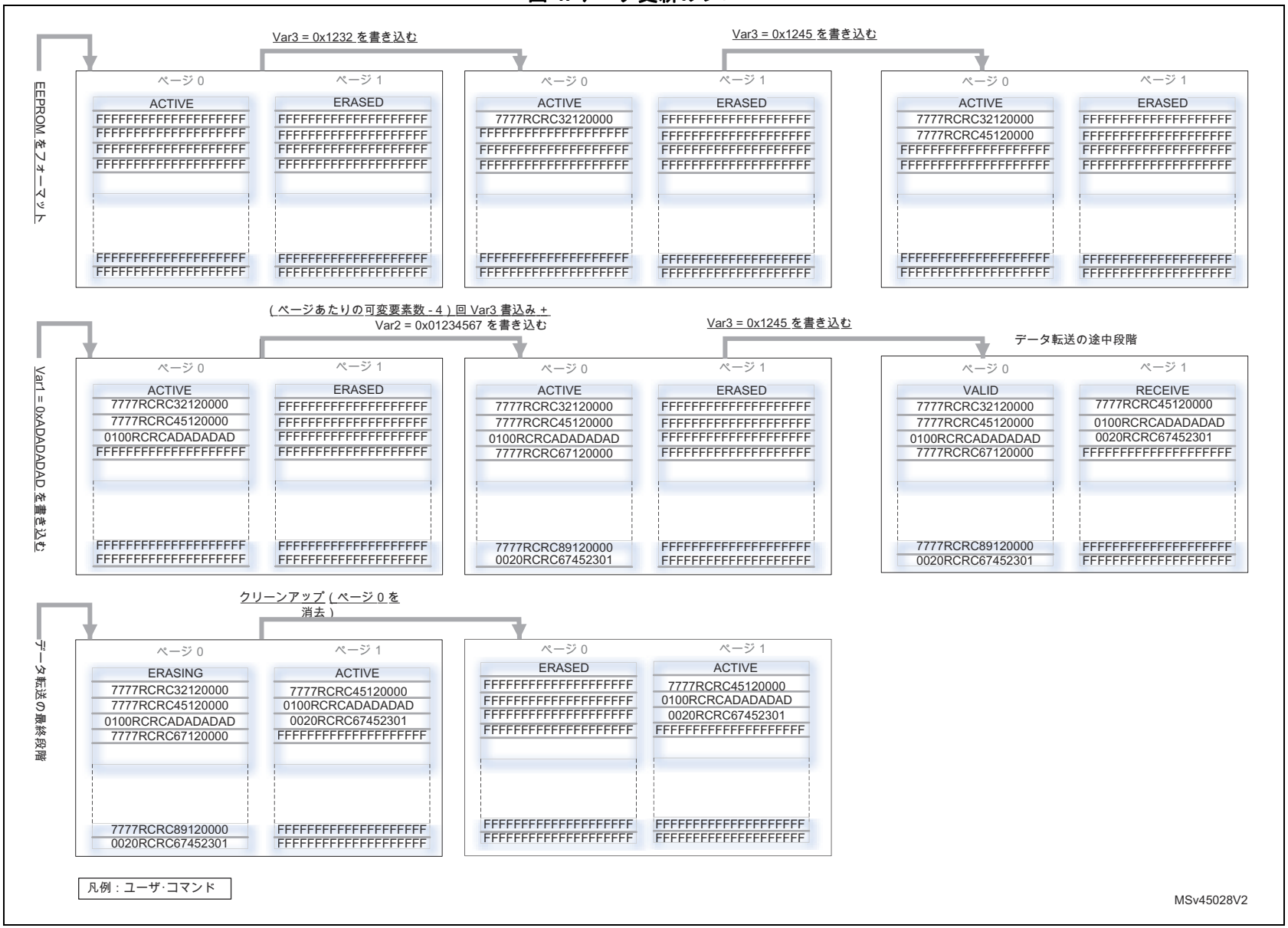
表 5. エミュレート EEPROM の仮想アドレス

可変要素名	仮想アドレス	データ幅
Var1	0x0001	32 bit
Var2	0x2000	32 bit
Var3	0x7777	16 bit

この例で必要な Flash ページは、ページ 0 とページ 1 の 2 ページだけです。図 4：データ更新のフローの RCRC は、この可変要素の 16 bit の CRC 値を表します（CRC に関する詳細は、セクション 4.6：EEPROM エミュレーションの堅牢性を参照してください）。

図 4：データ更新のフローには、書込みコマンドのみを示しています。書込みは、Flash アドレスの昇順にシーケンシャルに行われます。

図 4. データ更新のフロー



3.5 データ読出し

読出しコマンドは、ACTIVE または VALID ページの最高位アドレスから最低位アドレスに向けて Flash 読出しを実行し、有効なデータのみを返します。データが有効と見なされるのは、特定の仮想アドレスに最後に書き込まれた値であり、かつ CRC による整合性確認に合格している場合です。また、データ転送のメカニズムでは、有効なデータのみがコピーされることにも注意してください。

4 高度な機能

EEPROM エミュレーション·ファームウェアは、不揮発記憶領域という観点から、組込みアプリケーションのほとんどの要件を満たすように設計されています。このセクションでは、それらの要件の詳細を説明します。特定の場合には、他の組み込みアプリケーション要件が必要になる場合があり、それらについてはセクション 6：組込みアプリケーションの観点で取り上げます。

4.1 データ書換え単位の管理

エミュレート EEPROM は、バイト、ハーフワード、ワード単位で更新される不揮発のデータ記憶領域を必要とする組込みアプリケーションに使用できます。データ·サイズは通常、センサまたは通信インタフェースのデータ·サイズなど、アプリケーションの要件で決まります。

EEPROM エミュレーション·ファームウェアは、バイト、16 bit のハーフワード、32 bit のワード書換え単位に対応できるように設計されています。しかし、Flash メモリの使用率を最適化するために、ユーザ·アプリケーションによって、より小さなサイズのデータ要素をすべて 32 bit のデータ要素にまとめてから、その内容をエミュレート EEPROM に格納する方法を取ることも可能です。この方法により、16 bit の仮想アドレス、16 bit の CRC、32 bit のデータ値を同時に書き込めば、64 bit の Flash 行の最適使用が保証されます。

注： **Flash メモリの最小書込み幅は 64 bit です。これはECC（エラー訂正コード）機能がオフにできないためです。**

4.2 ウェアレベリング·アルゴリズムと Flash ページの割当て

ウェアレベリング·アルゴリズムを使用すると、Flash の書込みと消去動作を監視して、Flash のページ間でその回数を平準化できます。ウェアレベリング·アルゴリズムを使用しないと、ページの使用頻度にばらつきが発生します。例えば、寿命の長いデータが格納されているページは、頻繁に更新されるデータを含むページに比べて、書込み / 消去サイクルによる摩耗が少なくなります。ウェアレベリング·アルゴリズムは、Flash の各ページの使用可能なすべての書込みサイクル数を均等に使うようにします。

EEPROM エミュレーション·アルゴリズムは、Flash のページ間に書込み / 消去動作を均等に配分するように設計されています。Flash の書込みは、書き込むユーザ可変要素の種類に関係なく、アドレスの昇順にシーケンシャルに実行されます。1 つのページ·セットがフルになると、有効な要素が他のページ·セットにコピーされ、最初のページ·セットは完全に消去されます。

4.3 ガード･ページ

Flash ページの摩耗をさらに減らすために、ユーザには偶数の Flash ガード･ページを追加する選択も可能です（デフォルトではページ･セットあたり 1 ガード･ページ）。エミュレートされる可変要素数が少なく、ACTIVE ページに顕著な空き領域が残る場合には、ガード･ページは不要です。この数を増やすと（4、6、…）、Flash の書換え耐性を保証値よりも大きくできます。この機能は、エミュレート EEPROM の書換えサイクル機能と深く関連します。セクション 4.4：書換えサイクル能力：EEPROM 書換え耐性の向上を参照してください。

STM32L4 を使用する場合を例に取ると、1,000 個の EEPROM 可変要素を 4 Flash ページからなる 2 つのページ･セットに格納できます（各ページは 252 個の要素を保存できます）。すべての要素の書込みが 1 回終了した時点（またはページ転送後）では、新たなページ転送が開始されるまでに、残り 8 つの要素しか書き込むことができません。このような場合は、2 つのガード･ページを追加することを推奨します（ページ･セットあたり 1 ガード･ページ）。これによって、新たなページ転送が開始されるまでに 260 回の書込みを実行できるようになります。

4.4 書換えサイクル能力：EEPROM 書換え耐性の向上

EEPROM テクノロジーを使用する場合、1 バイト単位で個別に有限の回数（通常は1M（ 100万）回程度）だけプログラムできます。Flash テクノロジーの場合は、消去されていないアドレスには書き込むことができず、最小の消去サイズは Flash ページのサイズです。その結果、複数回の Flash 行への書込みアクセス後に 1 回の Flash ページ消去動作を実行するプログラム / 消去サイクルを定義する必要があります。

各STM32デバイスのオンチップフラッシュメモリページは、限られた回数だけ確実にプログラムおよび消去できます。各可変要素に必要な更新頻度が、この回数よりも多くなるような書込み集中型のアプリケーションの場合、ウェアレベリング･アルゴリズムを採用することで、エミュレート EEPROM の書換え耐性を向上させることができます。

表 6 に、このアプリケーション･ノートの対象製品に含まれるデバイスについて、高い信頼性でプログラムおよび消去動作が可能な回数の制限を示します。

表 6. Flash メモリの書換え耐性

STM32 シリーズ	Flash メモリの書換え耐性
STM32L4 シリーズ	10Kサイクル
STM32L4+ シリーズ	10Kサイクル
STM32L5 シリーズ	10Kサイクル
STM32G0 シリーズ	10Kサイクル（STM32G030x6/x8 と STM32G070CB/KB/RB については 1Kサイクル）
STM32G4 シリーズ	10Kサイクル
STM32WB シリーズ	10Kサイクル
STM32WL シリーズ	10Kサイクル

エミュレート EEPROM の必要サイズと、目標とする書換え耐性がわかれば、その目的に使用する Flash メモリのサイズを計算できます。Flash メモリ･サイズは、保存する可変要素のデータ幅の関数でもあります。

表 7. 4000 バイトのエミュレート EEPROM に対する Flash の使用量（STM32L4/L4+）

データ幅	10Kサイクルの書換え耐性に必要なページ数	10Kサイクルの書換え耐性のFlash サイズ	100Kサイクルの書換え耐性に必要なページ数	100Kサイクルの書換え耐性のFlash サイズ
STM32L4				
8 bit	34	68 KB	322	644(1) KB
16 bit	18	36 KB	162	324 KB
32 bit	10	20 KB	82	164 KB
STM32L4+				
8 bit	18	72 KB	162	648 KB
16 bit	10	40 KB	82	328 KB
32 bit	6	24 KB	42	168 KB

1. すべてのSTM32L4 には適用されません。EEPROM エミュレーションに割り当てることができるサイズは最大 512 KB です。各バンクのサイズについては STM32 製品のデータシートを参照してください。

注： **他の STM32 シリーズについては、セクション 4.5 : EEPROM エミュレーションに必要な Flash サイズの計算に示した数式を適用すれば、上表と同じ情報を得ることができます。**

4.5 EEPROM エミュレーションに必要な Flash サイズの計算

最初の近似として、必要な Flash メモリのサイズがエミュレート EEPROM のサイズおよび書換えサイクル能力に比例するものとします。STM32L4 シリーズと STM32L4+ シリーズのマイクロコントローラについて、保存する要素のデータ幅に応じて必要な Flash メモリのサイズを見積るには表 7：4000 バイトのエミュレート EEPROM に対する Flash の使用量（STM32L4/L4+）を使用できます。

Flash メモリのサイズ要件を厳密に求めるには、STM32 のシリーズにかかわらず、次式を使用できます。

$$\text{Flash サイズ} = \left\lceil \left\lceil \frac{\text{EEPROM 可変要素数}}{\text{ページあたりの可変要素数}} \right\rceil \times \frac{\text{サイクル数}}{\text{Flash メモリ書換え耐性 (kcycles)}} \times 2 + \text{ガード·ページ数} \right\rceil \times \text{ページ·サイズ}$$

ここで、$\lceil \ \rceil$ は、囲まれた数値以上の最小の整数を表します。

注： **該当する STM32 シリーズの Flash メモリ書換え耐性の値（Kサイクル）を表 6：Flash メモリの書換え耐性に示します。**

この式は次の手順に分解できます。

1. 1 つの Flash ページが保存できる要素数（表 4：Flash メモリのプロパティ参照）を把握し、EEPROM 要素の保存に必要な Flash ページ数を計算します。（計算される Flash メモリ·サイズは、要素のデータ幅には依存しません。）
2. このページ数に、目標とするサイクル比（整数）を乗じます（サイクル比は、例えば、Flash メモリの書換え耐性が 10Kサイクルの場合、標準の 10Kサイクルの書換え耐性を目指すならば 1、100Kサイクルの書換え耐性の場合は 10 とします）。製品の Flash メモリ書換え耐性（表 6：Flash メモリの書換え耐性参照）を考慮する必要があります。
3. データ転送メカニズムを考慮して、この Flash ページ数を 2 倍します。
4. これにガード·ページ数（偶数）を加算すれば、最終的な Flash ページ数が得られます。
5. 使用する製品の Flash ページ·サイズを把握して（表 4：Flash メモリのプロパティ参照）、EEPROM エミュレーションに必要な Flash メモリ·サイズを計算します。

STM32L4 の計算例

4000 バイトを個別に保存するには、各ページが最大 252 個の要素を保存できることから、1 つのページ·セットは 16 の Flash ページで構成する必要があります。この第 1 のページ·セットがフルになったときにデータを転送できるように、同じサイズの第 2 のページ·セットを用意する必要があります。ガード·ページが 2 ページ必要であると仮定すると、34 ページの Flash ページが必要になります。

注： **この計算は、若干控えめな見積りです。**

4.6 EEPROM エミュレーションの堅牢性

組込みアプリケーションでは、Flash メモリのプログラムまたは消去中に電源障害や非同期リセットが発生する可能性があります。その場合、Flash 行（プログラムの場合）または Flash ページ全体（消去の場合）の内容が不明になります。これはエラー訂正コードによる検出（ECCD）または訂正（ECCC）の有無にかかわらず、シングルビットまたは複数ビットのエラーをもたらす可能性があります。

注： **ハードウェアによる ECC は、Flash メモリの消耗後のエラーを検出、訂正するために設計されており、Flash のプログラムや消去動作の中断を確実に検出するようには設計されていません。**

EEPROM エミュレーション･ソフトウェアは、電源障害や非同期のリセットに対して高い堅牢性を持つように設計および検証されています。この堅牢性は、次節で述べる Flash インタフェースと EEPROM エミュレーション･ドライバが持ついくつかの機能によって実現されています。Flash の制約や問題の回避策については、製品のエラッタ・シートを参照してください。

4.6.1 データ･リカバリ

壊れたデータデータ（仮想アドレスまたはデータの値、あるいは、それら両方）を検出するために、16 bit の CRC（巡回冗長検査）を実装しました。これは、次の多項式による ANSI CRC-16 に基づいています：$x^{15} + x^2 + 1$（これを 0x8005 と表記します）。この多項式は極めて高い検出率を備えていますが、EEPROM 設定ファイルによりユーザが変更することも可能です。

可変要素を書き込む場合、対応する CRC の値も併せて保存されます。可変要素の読出しまたは転送を実行する際、CRC を計算し、その可変要素に保存されていた値と比較して検証します。一致すれば、その可変要素は有効と見なされます。不一致の場合、可変要素は無効化されます。

注： **EEPROM エミュレーション･ソフトウェアでは CRC の計算を高速化するために CRC ペリフェラルを使用しています。**

データ･リカバリ機能は、既にプログラム済みのヌル以外のデータを含むワードに書き込めるのは完全なゼロ（64 bit すべてがゼロ）だけであるという事実に基づいており、可変要素の無効化は、そこにゼロを書き込むことで実行します。このため、可変要素の仮想アドレスには 0x0000 を使用できません。

注： **仮想アドレス 0xFFFF も禁止です。こちらは消去済みの Flash 行に対応するからです。**

データ･リカバリのメカニズムでは、この Flash 行に対して ECCD の NMI（ノンマスカブル割込み）がトリガされた場合にも、ユーザがゼロを書き込む必要があります。これは、ECC によって訂正不可能なエラーが検出された場合（つまり ECCD ビットがセットされた場合）、`FI_DeleteCorruptedFlashAddress` 関数を呼び出すことで実現します。

このメカニズムは、`EE_Init` 関数でのクリーンアップ・フェーズが実行されている間だけ適用されます。このフェーズが完了すると、これ以上 NMI をトリガするイベントは存在しないものと仮定されます。

ゼロをプログラムする動作が何らかの原因で失敗し、Flash プログラムのエラー･フラグ（PROGERR、PGAERR、PGSERR）が設定される場合があります（例えば、Flash 行が既に 0 だった場合、PROGERR フラグが設定されます）。この場合、その行は無効と見なされ、現在の状態が保たれます（NMI サービス･ルーチンは何の動作も行わずに終了します）。

PROGERR、PGAERR、PGSERR 以外のフラグが設定された場合、NMI ルーチンは無限ループに入ります。

クリーンアップ・フェーズにもかかわらず、EEPROM エミュレーションにおいて他の NMI がトリガされた場合、何の動作も実行されません。NMI がトリガされたという事実は、行が無効化されたものと見なされます。クリーンアップ・フェーズにより、非同期リセットや電源障害により壊れた行の大半は除外されます。したがって、クリーンアップ・フェーズの後、残りのNMIの数はほとんどの場合0になります。

ある特定のアドレスの可変要素への最初の書込み時に、書込み動作が電源障害か非同期リセットによって中断された場合、ドライバは、このアドレスにはデータが存在しないものと見なします。それ以外のすべての場合、EEPROM エミュレーション·ソフトウェアは常に、Flash メモリに保存されている以前の値を見つけ出して、これを最新の有効データ値として返します。

4.6.2 ページ·ヘッダのリカバリ

データ破壊と同様に、ページ·ヘッダも、ヘッダの更新または Flash ページの消去中に発生する電源障害や非同期リセットによって破壊される可能性があります。

この破壊の検出とリカバリのために、`EE_Init` ルーチンが実装されています。このルーチンは電源投入後、ただちに呼び出す必要があります。一連のページ・ステータスについて整合性を確認し、必要に応じて修復します。これによって、データ損失を確実に防止します。詳細については、セクション 5.3 : EEPROM エミュレーションのタイミングを参照してください。

発生する可能性がある障害のシナリオを回避するために、消去済みのページも、リセット時に系統的に再消去を実行します（`EE_Init` ルーチン内で）。これは安全な動作を保証する一方、エミュレート EEPROM の書換えサイクル能力は消費します。Flash の書込みまたは消去中に非同期リセットや電源障害が発生しないようにアプリケーションを設計すれば、この系統的消去の必要性は回避できます。詳細については、セクション 6.2 : 電源障害の検出を参照してください。

堅牢性の実現には、`EE_Init` ルーチンに実装される、いくつかのチェックとリカバリのメカニズムという代償を伴います。代償とは、より単純な EEPROM エミュレーションに比べて、コード·サイズの若干の増加や初期化に要する時間の増大を招くことです。詳細は、セクション 5.3 : EEPROM エミュレーションのタイミングを参照してください。

4.7 リアルタイム性に関する考慮事項

付属の EEPROM エミュレーション·ファームウェアは、内蔵 Flash メモリを使用して動作するように実装されています。したがって、Flash の消去またはプログラム動作中（EEPROM の初期化、可変要素の更新、ページ消去の間）は、同じ Flash バンクへのアクセスがストール（停止）します。

そのため、アプリケーション·コードが実行されず、割込みの処理も一定時間、行われません。この時間の最大値は、使用する STM32 製品のページ消去動作時間の最大値に等しくなります（該当するデータシートを参照してください）。こうした動作は、多くのアプリケーションでは許容できますが、リアルタイム性という制約のあるアプリケーションの場合は、ユーザが是正措置を講じる必要があります。

4.7.1 RWW（Read While Write）機能付き Flash メモリを内蔵したデバイス

デュアルバンクの Flash メモリを内蔵した STM32 デバイスでは、重要なルーチン（ベクタ·テーブル、重要な割込みサービス·ルーチン）を一方のバンクに配置し、EEPROM エミュレーションに使用する領域はもう一方のバンクに配置することを推奨します。その場合も EEPROM エミュレーションに使用するバンクの Flash 領域を使用できますが、Flash の書込みと消去の動作中は実行が遅延します。

注 : 使用する STM32 デバイスが Flash RWW 機能に対応しているかどうかは、製品のデータシートを参照してください。

4.7.2 内部 RAM からの重要プロセスの実行

リアルタイム性の制約を満たすもう一つの方法は、重要なコードを内部 RAM から実行します。

1. ベクタ·テーブルを内部 RAM に移動します。
2. すべての重要なプロセスや割込みサービス·ルーチンを内部 RAM から実行します。コンパイラには、関数を RAM 上に配置するキーワードが用意されています。関数は、システムの起動時に、他の初期化される変数と同様に Flash メモリから RAM にコピーされます。

注： **RAM 関数では､使用するデータおよび呼び出される関数のすべてを RAM 内に配置する必要があることに注意してください。**

4.7.3 デュアルコアに関する考慮事項

STM32 デバイスの中には、2 つのコアを使用できるものがあります（詳細は、該当する STM32 製品のデータシートやリファレンス·マニュアルを参照してください）。例えば、STM32 ワイヤレス·シリーズ製品の*大多数*が、これに当てはまります。

STM32WB シリーズのマイクロコントローラは、メイン·コア（CPU1、ARM Cortex-M4）に加えて、無線専用の第 2 のコア（CPU2、ARM Cortex-M0+）を搭載しています。ここでは、例として無線用途のシナリオについて検討します。このシナリオでは CPU1 が EEPROM エミュレーションを実行し、Flash 動作を処理する必要があります。

これを CPU2 との競合を避けながら実現するには､CPU2 上で動作中の無線アプリケーションについて、CPU1 側で以下の項目を確認する必要があります。

- 現在、Flash メモリにアクセス中でないこと
- Flash メモリへの緊急のアクセスを必要としないこと

注： **特に消去動作は、無線プロトコルの仕様上、比較的長い時間を必要とします。したがって、CPU2 が緊急の Flash 動作を既に計画している場合は、CPU1 の長い動作（消去など）の開始を避けるメカニズムの実装も必要になります。**

EEPROM エミュレーション·アプリケーションをデュアルコア製品に適用するアプリケーション開発者は、上記のような検証を実装する必要があるかも知れません。

例えば、STM32WB シリーズのマイクロコントローラの場合は、次のような機能を使用できます。

- ハードウェア·セマフォ機能（HSEM）
- PWR_EXTSCR レジスタ·ビットの CRPF ビット（重要な無線動作の実行中にセットされます）

どちらも、STM32WB シリーズのマイクロコントローラで使用できます。

注： **デュアルコア·アプリケーションの中には、無線以外にも、これと同様の課題を伴うものがあります。**

Bluetooth® Low Energy（BLE）スタックを内蔵した STM32WB シリーズ製品で、主にハードウェア·セマフォ機能によって実現している Flash アクセス管理メカニズムについては、アプリケーション·ノート [2] で解説しています。

P-NUCLEO-WB55 Nucleo ボードの場合、X-CUBE-EEPROM ファームウェア·パッケージに、次の 2 つの例が付属しています（セクション 5.1.2：STM32Cube 拡張ソフトウェア（X-CUBE-EEPROM）参照）。

- 単純に EEPROM エミュレーション·ドライバを活用するための例（他のシリーズ向けのサンプルと同じ）
- EEPROM 動作の処理中に BLE の接続と通信を維持する例これは、AN5289 [2] に記載された Flash へのアクセス·タイミングのルールに従っており、このメカニズムを実装する際の参考になるでしょう。

RF スタックを備えた他の ST マイクロコントローラ向けに、上記のメカニズムを利用できるかどうかを判断するには、それらの製品のデータシートや他の関連ドキュメントを参照してください。

4.8 割込みまたはポーリング·モードによる Flash メモリのクリーンアップ

有効な要素の新しいページ·セットへの転送が完了したら、前のページ·セットを消去する必要があります。EEPROM エミュレーション·ドライバは、メイン·アプリケーションに対して Flash メモリのクリーンアップを要求するフラグを立てます。アプリケーション·プログラム側では、アプリケーションのリアルタイム性の制約が緩和されるまで（例えば、低電力モードに移行する前など）、このクリーンアップを遅延させることができます。

Flash のクリーンアップは、ポーリング·モードで実行できます（`EE_Cleanup` 関数の呼び出しによる）。これは、EEPROM エミュレーションに使用している Flash ページ·セットの 1 つで消去が完了するまでクリーンアップ関数がリターンしないことを意味します。割込みモードのクリーンアップも可能です（`EE_Cleanup_IT` 関数の呼び出しによる）。この場合、クリーンアップ関数は即座にリターンし、その後のページ消去は、後続の割込みサービス·ルーチンによって実行されます。

4.9 キャッシュ·コヒーレンシのメンテナンス

ほとんどの STM32 マイクロコントローラ·シリーズでは、アプリケーション実行中に Flash メモリの内容が変更されることを想定していません。したがって、一部の STM32 マイクロコントローラでは、Flash のデータをキャッシュし、その後これが変更されても、キャッシュはそれを無視するので、キャッシュ·コヒーレンシが失われます（キャッシュ内のデータと、Flash メモリ内の新しい値が一致しなくなります）。X-CUBE-EEPROM ファームウェアには、キャッシュ·コヒーレンシを維持するためのコードが実装されているため、ユーザは必要に応じてキャッシュをアクティブ化できます（命令キャッシュ、データ·キャッシュ、プリフェッチ·バッファなど。どのようなキャッシュが提供されているかは、使用する製品のリファレンス·マニュアルを参照してください）。

例えば、STM32L5 シリーズの X-CUBE-EEPROM の例では、MPU（メモリ保護ユニット）を使用して、EEPROM Flash 領域をキャッシュ不可と定義しています。もう一つの例は STM32L4 シリーズです。製品のリファレンス·マニュアルの提言に従い、ページ消去動作後に DCache を一掃しています。

5 API およびアプリケーション例

5.1 EEPROM エミュレーション・ソフトウェアの説明

このセクションでは、使用するデバイス・シリーズに対して ST マイクロエレクトロニクスが提供する STM32Cube ファームウェアを使用して、EEPROM エミュレーションを実装するドライバについて説明します。

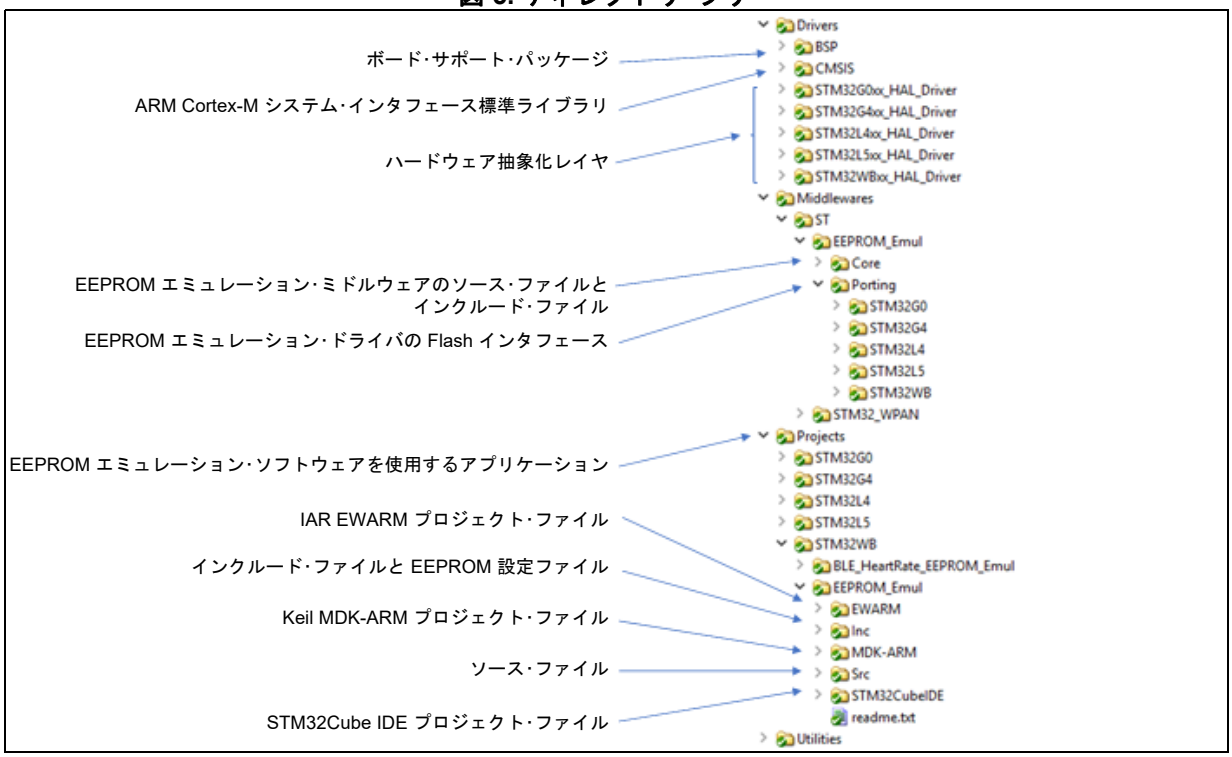
5.1.1 主な特徴

- ユーザ設定可能な EEPROM サイズ
- 8、16、32 bit の可変要素をサポート
- EEPROM メモリの書換え耐性を Flash メモリの書換え耐性よりも向上
- ウェアレベリング・アルゴリズム
- プログラムおよび消去動作中の割込み処理が可能
- 非同期リセットや電源障害に対する堅牢性
- マルチコア STM32 デバイス（STM32WB シリーズ製品など）においてコア間で共有される Flash メモリを保護する機能の実装（任意）

5.1.2 STM32Cube 拡張ソフトウェア（X-CUBE-EEPROM）

EEPROM エミュレーション・ソフトウェアは、X-CUBE-EEPROM と呼ばれる専用ソフトウェア・パッケージとして提供されます。すべての開発ボードおよび製品シリーズに準拠するために、ミドルウェア・ユーティリティに分類されます。

図 5. ディレクトリ・ツリー



1,000 個の不揮発可変要素を管理する方法の実例として、EEPROM エミュレーション・ドライバを使用したサンプル・デモ・プロジェクトも提供されています。ボード用に提供されているサンプル・プロジェクトの一覧を表 8 に示します。これらは、パッケージのフォルダである *Projects¥STM32xx¥EEPROM_Emul* に保存され、同じ STM32 マイクロコントローラ・シリーズを使用する他のボード向けに簡単にカスタマイズできます。

P-NUCLEO-WB55.Nucleo ボード用にもう 1 つの例が付属しています。それは、*Projects¥STM32WB¥BLE_HeartRate_EEPROM_Emul¥P-NUCLEO-WB55.Nucleo* に保存されています。Nucleo ボードは、EEPROM 動作の処理に加えて BLE の接続と通信を維持します。DUALCORE_FLASH_SHARING プリプロセッサ定義シンボルを使用することで、この例はセクション 4.7.3：デュアルコアに関する考慮事項で説明した Flash の共有メカニズムやタイミングのルールに従っています。

表 8. 対象ボード

STM32 シリーズ	対象ボード
STM32L4 シリーズ	STM32L476G-Discovery
STM32L4+ シリーズ	STM32L4R5ZI_Nucleo
STM32L5 シリーズ	NUCLEO-L552ZE-Q
STM32G0 シリーズ	NUCLEO-G071RB
STM32G4 シリーズ	NUCLEO-G431RB、NUCLEO-G474RE
STM32WB シリーズ	P-NUCLEO-WB55
STM32WL シリーズ	NUCLEO-WL55JC

プロジェクトには次の 4 つのソース・ファイルがあります。

- ***eeprom_emul.c*** ：ユーザ・プログラムから呼び出し可能な以下の EEPROM エミュレーション・ファームウェア関数が含まれます。

  `EE_Format`
  `EE_Init`
  `EE_ReadVariable32bits`
  `EE_WriteVariable32bits`
  `EE_ReadVariable16bits`
  `EE_WriteVariable16bits`
  `EE_ReadVariable8bits`
  `EE_WriteVariable8bits`
  EE_CleanUp
  EE_CleanUp_IT
  EE_DeleteCorruptedFlashAddress

- ***flash_interface.c*** ：STM32 の Flash に固有の機能に対応するために必要な関数が含まれます。
- ***main.c*** ：このアプリケーション・プログラムは、エミュレート EEPROM の設定、書込み、読出しを行うために、本項で説明しているルーチンを使用した例です。
- ***stm32xxxx_it.c*** ：NMI ハンドラで `EE_DeleteCorruptedFlashAddress` 関数を使用する、割込みサービス・ファイルの例です。

BLE STM32WB シリーズの例では、ドライバの活用法を示すためのコード `BLE_HeartRate_EEPROM_Emul` がファイル *Core/Src/app_entry.c* に記述されています。特に、`EEPROM_Emul_Init` と `EEPROM_Emul_Operation` 関数は、EEPROM ドライバの初期化と使用（読出しと書込みの動作）のためのシーケンサのタスク例です。

5.1.3 ユーザ定義パラメータ

EEPROM エミュレーション·アルゴリズムのパラメータは、アプリケーションの要件に従って設定する必要があり、ファイル eeprom_emul_conf.h 内に記述されています。

- **NB_OF_VARIABLES**（デフォルト値 1000）: 不揮発要素の数。各要素の値は 8、16、32 bit のいずれかです。
- **START_PAGE_ADDRESS**: EEPROM シミュレーションに使用する最初の Flash ページのアドレス。
  デュアルバンク対応のデバイスで、その特徴のメリットを生かすために推奨されるこのパラメータの値の設定例は、第 2 のバンクの最初のアドレスです（セクション 4.7.1 : RWW（Read While Write）機能付き Flash メモリを内蔵したデバイス参照）。
- **CYCLES_NUMBER**（デフォルト値 1）: EEPROM の等価書換え耐性を表す、Kサイクル数（X）の数。CYCLES_NUMBER が 10 の場合、エミュレート EEPROM は 10 x X Kサイクル相当の書換え耐性を持ちます。X は使用する製品の Flash 書換え耐性で、表 6 : Flash メモリの書換え耐性に一覧を示してあります。
- **GUARD_PAGES_NUMBER**（デフォルト値 2）: Flash メモリへの負担を軽減するために使用するガード·ページの数。値は偶数である必要があります。
- **CRC_POLYNOMIAL_LENGTH**（デフォルト値 16）: ほとんどの場合、変更は不要です。16 bit の CRC は計算速度と Flash サイズの観点から最適であり、極めて良好な検出率も確保できます。
- **CRC_POLYNOMIAL_VALUE**（デフォルト値 0x8005）: ほとんどの場合、変更は不要です。ANSI CRC は計算速度と Flash サイズの観点から最適であり、極めて良好な検出率も確保できます。

STM32WB シリーズの Bluetooth Low Energy スタックに関連してデュアルコア Flash 共有メカニズムをアクティブ化するには、ユーザ·ソフトウェアで DUALCORE_FLASH_SHARING をプリプロセッサ定義シンボルとして定義する必要があります。

5.1.4 ユーザ API の定義

アプリケーション·プログラムによって呼び出し可能な一連の関数を表 9 に示します。これらの定義は *eeprom_emul.c* モジュール内に記述されています。

表 9. API 定義

関数名	内容
EE_Format	EEPROM エミュレーションに使用する全 Flash ページを消去し、最初のページに ACTIVE ヘッダを書き込みます。
EE_Init	EEPROM エミュレーションの可変要素を初期状態に設定するとともに、Flash の書込みまたは消去動作中に非同期リセットや電源障害が発生した場合には、既知の有効な状態に Flash ページを復元します。 消去が必要な Flash ページ（ERASED 状態で完全に消去されていないページ、ERASINGステータスのページなど）を消去します。デフォルトでは EE_FORCE_ERASE パラメータを使用する必要があります。アプリケーションとして、Flash の書込みまたは消去動作中に非同期リセットおよび電源障害のいずれも発生しないと保証できる場合は EE_CONDITIONAL_ERASE パラメータを使用できます。セクション 4.6.2 : ページ·ヘッダのリカバリを参照してください。 リセット後、エミュレート EEPROM にアクセスする前に、この関数を系統的に呼び出す必要があります。
EE_ReadVariableXXbits	この関数は、パラメータとして渡された仮想アドレスに対応するデータ値を更新します。最後に保存された要素のみが読み出されます。 与えられた仮想アドレスにデータが一切見つからなかった場合を除き、EE_OK のステータスを返します。 サポート対象のデータ·サイズ（XX = 8、16、32）のすべてに対応できるように 3 つの関数が実装されています。
EE_WriteVariableXXbits	この関数は、EEPROM の特定仮想アドレスの内容を、パラメータとして渡されたデータ値で更新します。 EEPROM エミュレーションに使用するページ·セットがフルの場合、この関数が Flash ページの転送を開始し、EE_CLEANUP_REQUIRED のステータスを返します。 NUCLEO-WB55 Nucleo ボードの BLE アプリケーション（DUALCORE_FLASH_SHARING が定義されているアプリケーション）では、Flash が CPU2 によって使用されている場合に EE_FLASH_USED を返すことができます。(1) サポート対象のデータ·サイズ（XX = 8、16、32）のすべてに対応できるように 3 つの関数が実装されています。
EE_CleanUp	ERASINGステータスにあるページ·セットを消去します。アプリケーション·プログラムは、リアルタイム性の制約が低いときにこの関数を呼び出すことができます。Flash のクリーンアップがポーリング·モードで実行されるため、この関数は EEPROM エミュレーションに使用する Flash ページ·セットの 1 つの消去が完了するまでリターンしません。
EE_CleanUp_IT	ERASINGステータスにあるページ·セットを消去します。Flash クリーンアップが割込みモードで開始されるため、この関数は即座にリターンし、その後のページ消去は後続の割込みサービス·ルーチンによって実行されます。
FI_DeleteCorruptedFlashAddress	この関数は、NMI に基づいて呼び出すことが可能で、Flash ECCD で検出された壊れた Flash アドレスを削除して、この Flash インタフェース障害をクリアします（セクション 4.6.1 : データ·リカバリ参照）。

1. EE_WriteVariableXXbits が EE_FLASH_USED を返せるのは、EEPROM 可変要素への直接書込みの場合のみであり、ページ転送、ページ·ヘッダ更新、初期化プロセスについては対応していません。また、この値が返された場合、ドライバは HSEM 割込みをアクティブ化することで、CPU2 が Flash デバイスの使用に関連するセマフォを解放したときに割込みを生成できます。こうすれば、ユーザ·プログラムはセマフォが解放されるのを待つ間に、他の処理を実行することができます。

5.2 EEPROM エミュレーションのメモリ·フットプリント

表 10 に、Flash メモリ、RAM、スタック·サイズの観点から見た EEPROM エミュレーション·ドライバのフットプリントの詳細を、このアプリケーション·ノートの対象製品の一部について示します。

ここに示す数値は、サイズ最適化を「高」レベルに設定した IAR EWARM 8.42.2 ツールを使用して決定したものです。このアルゴリズムはヒープを使用しないことに注意してください。

表 10. EEPROM エミュレーション·メカニズムのメモリ·フットプリント

STM32 シリーズ	必要なコードとデータのサイズ（バイト単位）(1)	
	Flash	SRAM
STM32L4 シリーズ	4256	12
STM32L4+ シリーズ	4286	12
STM32L5 シリーズ	4728	12
STM32G0 シリーズ	3812	12
STM32G4 シリーズ	4946	12
STM32WB シリーズ	5392	16
STM32WB シリーズ(2)	3328	14

1. 要素 1,000 個の場合の値（16 bit アドレス、16 bit CRC、32 bit データ）
2. BLE 通信を伴う例

表 10 は、EEPROM エミュレーション自体に使用する Flash のサイズは考慮していません。対応する Flash メモリ·サイズを計算するには、セクション 4.5 : EEPROM エミュレーションに必要な Flash サイズの計算に示した数式を参照してください。

EEPROM 要素の値を保存するために、アプリケーション·プログラム（*main.c*）が変数 `VarDataTab[]` を実装していることにも注意してください。32 bit の要素 1,000 個に対応する場合、そのサイズは 4000 バイトになります。

柔軟性を確保するために、付属する例では、この変数を RAM に配置しています。さらに、ほとんどのユーザ·アプリケーションでは Flash メモリに保存されている要素を RAM にコピーしたものを必要としないため、`VarDataTab[]` は簡単に削除できます。

5.3 EEPROM エミュレーションのタイミング

このセクションでは、1,000 個の 32 bit 可変要素を扱う EEPROM エミュレーション・ドライバのタイミング・パラメータについて説明します。

すべてのタイミングは、次の条件の下で計測します。

- VDD = 3.3 V
- Flash メモリからの実行（デュアルバンク使用時は EEPROM 可変要素に使用していない方のバンク）
- 室温
- 動作の各種類を 100 回実行

表 11 に、上記の条件で測定した EEPROM エミュレーション・ドライバのタイミング値を、本アプリケーション・ノートの対象製品の一部について示します。異なる条件の下では（可変要素の数、書換え耐性のサイクル数、マイクロコントローラ周波数などが異なる場合）、これらのタイミングとは大幅に異なる値になる可能性があります。

表 11. 各種ターゲットの EEPROM エミュレーションのタイミング

ターゲット	条件	動作	平均値	最小値	最大値
STM32L476G-DISCO	– システム・クロック = 80 MHz – ART 有効 – Flash プリフェッチ無効 – 書換え耐性 = 100Kサイクル	強制消去による EEPROM 初期化 (1)	1.2215 s	904.4 ms	2.0212 s
		条件付き消去による EEPROM 初期化 (1) (2)	365.56 ms	6.1 ms	1.2499 s
		EEPROM への書込み動作	注 (3) を参照	90 µs	344.90 ms
		EEPROM からの読出し動作 (4)	47.412 µs	9.3 µs	311.5 µs
		EEPROM のクリーンアップ (1)	900.27 ms	900.13 ms	900.38 ms
		EEPROM のクリーンアップ（割込み） (1)	900.19 ms	900 ms	900.38 ms
NUCLEO-L4R5ZI	– システム・クロック = 80 MHz (5) – ART 有効 – Flash プリフェッチ無効 – 書換え耐性 = 100Kサイクル	強制消去による EEPROM 初期化 (1)	741.69 ms	509.6 ms	884 ms
		強制消去による EEPROM 初期化 (1) (2)	466.91 ms	5.8 ms	776.9 ms
		EEPROM への書込み動作	注 (3) を参照	90 µs	307.20 ms
		EEPROM からの読出し動作 (4)	75.639 µs	9 µs	314.3 µs
		EEPROM のクリーンアップ (1)	461.90 ms	461.8 ms	462 ms
		EEPROM のクリーンアップ（割込み） (1)	461.86 ms	461.8 ms	462 ms

表 11. 各種ターゲットの EEPROM エミュレーションのタイミング

ターゲット	条件	動作	平均値	最小値	最大値
NUCLEO-L552ZE-Q	– システム・クロック = 110 MHz – ICACHE 無効 – 書換え耐性 = 10Kサイクル	強制消去による EEPROM 初期化 (1)	365.10 ms	111.1 ms	670 ms
		条件付き消去による EEPROM 初期化 (1) (2)	267.5 ms	1.5 ms	680 ms
		EEPROM への書込み動作	注 (3) を参照	90 µs	580.18 ms
		EEPROM からの読出し動作 (4)	149.3 µs	10 µs	485.6 µs
		EEPROM のクリーンアップ (1)	109.85 ms	109.80 ms	109.89 ms
		EEPROM のクリーンアップ (割込み) (1)	109.85 ms	109.81 ms	109.88 ms
NUCLEO-G071RB	– システム・クロック = 64 MHz – 書換え耐性 = 10Kサイクル	強制消去による EEPROM 初期化 (1)	251.17 ms	111.4 ms	814.4 ms
		条件付き消去による EEPROM 初期化 (1) (2)	233.43 ms	1.47 ms	792.9 ms
		EEPROM への書込み動作	注 (3) を参照	90 µs	702.18 ms
		EEPROM からの読出し動作 (1)	102.16 µs	14 µs	592 µs
		EEPROM のクリーンアップ (1)	110.39 ms	110.37 ms	110.40 ms
		EEPROM のクリーンアップ (割込み) (1)	110.41 ms	110.40 ms	110.43 ms
NUCLEO-G431RB	– システム・クロック = 170 MHz – 書換え耐性 = 10Kサイクル	強制消去による EEPROM 初期化 (1)	179.33 ms	110.3 ms	349.4 ms
		条件付き消去による EEPROM 初期化 (1) (2)	160.90 ms	600 µs	356.3 ms
		EEPROM への書込み動作	注 (3) を参照	90 µs	260.67 ms
		EEPROM からの読出し動作 (4)	74.242 µs	6 µs	170 µs
		EEPROM のクリーンアップ (1)	109.89 ms	109.86 ms	109.89 ms
		EEPROM のクリーンアップ (割込み) (1)	109.90 ms	109.89 ms	109.92 ms
P-NUCLEO-WB55.Nucleo (6)	– システム・クロック = 64 MHz – 書換え耐性 = 10Kサイクル	強制消去による EEPROM 初期化 (1)	276.83 ms	67.27 ms	561.32 ms
		条件付き消去による EEPROM 初期化 (1) (2)	258.65 ms	1.86 ms	539.78 ms
		EEPROM への書込み動作	注 (3) を参照	90 µs	476.26 ms
		EEPROM からの読出し動作 (4)	289.25 µs	11 µs	570 µs
		EEPROM のクリーンアップ (1)	65.867 ms	65.82 ms	65.92 ms
		EEPROM のクリーンアップ (割込み) (1)	64.966 ms	64.90 ms	65.92 ms

1. 初期化とクリーンアップの最大時間は、使用する Flash EEPROM ページの数と、1 ページの消去にかかる時間で決まります。(STM32 製品データシートの「Flash メモリの特性」のセクションを参照してください。) Flash EEPROM ページの消去に要する時間 = ページ数 * tERASE

2. EEPROM エミュレーションが非同期リセットや電源障害によって中断されないことをアプリケーションが保証している場合、条件付き消去を使用できます。この方法は、実行時間だけでなく EEPROM の書換え耐性の面からも高い効率が得られます (詳細はセクション 4.6.2 : ページ・ヘッダのリカバリを参照してください)。

3. 書込み時間の代表値は、最小書込み時間に近い値になります。ほとんどの場合、データ転送を伴わないためです。最大値は、データ転送が発生する書込み動作に対応しています。

4. 最小値は、Flash メモリに保存されている最後の要素の読出し動作に対応し、最大値は Flash メモリに保存されている最初の要素の読出し動作に対応しています。
5. STM32L4+ シリーズのボード（NUCLEO-L4R5ZI）のタイミング測定は、STM32L4 シリーズ（NUCLEO-L476RG）と比較できるように、80 MHz のシステム･クロックを使用して行いました。ただし NUCLEO-L4R5ZI は、クロック周波数を最大 120 MHz まで高速化できるのに対し、NUCLEO-L476RG は上限が 80 MHz です。他のボードの測定は、最大システム･クロック周波数で行っています。
6. BLE 機能なしの例から取得したデータ

注：　これらのタイミング測定は、読者が各種 STM32 シリーズによるエミュレーション･ファームウェアの性能の感触を得られるようにすることを目的に提供するものです。動作の各種類を 100 回実行して測定した値であるため、絶対的な基準値ではありません。最小値、最大値、場合によっては平均値でさえ異なる値が示される可能性があります（特に測定条件が変更されている場合）。

6 組込みアプリケーションの観点

このセクションでは、組込みアプリケーションにおけるソフトウェアの制約を克服する方法や、各種アプリケーションのニーズを満たす方法に関するアドバイスを提供します。

6.1 データ保持期間

Flash メモリのデータ保持期間の代表値は、1,000 サイクルの書換え後、85 ℃で 30 年間です（より完全なデータは STM32 製品のデータシートを参照してください）。EEPROM エミュレーション·ドライバでは、EEPROM のデータ保持期間を延長するための特別な措置は講じていません。しかし、ページ転送が開始されると、すべてのデータ（長寿命のデータを含む）が新しい Flash ページにコピーされます。この動作により、EEPROM のデータ保持期間は、元の Flash データ保持期間に比べて著しく延長されます。

6.2 電源障害の検出

アプリケーションにより非同期リセットが生成される可能性がない場合、Flash のプログラムまたは消去動作が阻害されるのは電源障害が発生した場合に限られます。アプリケーションで PVD（プログラム可能な電圧検出器）を使用すれば、VDD のランプ·ダウン（電圧降下）が発生している間は Flash の書込みや消去動作が発生しないようにすることができます。EEPROM のメイン·アプリケーション例には、この PVD の使用例が含まれています。

書込みまたは消去の全動作期間中、および電源電圧が 1.7 V（最小動作電圧）を下回るか、BOR（ブラウンアウト·リセット）が生成されるまでは、十分な電力をチップに供給できるだけの容量を備えたデカップリング·コンデンサまたはバッテリが必要です。このような条件が満たされている場合は、`EE_Init` ルーチンの `EE_CONDITIONAL_ERASE` パラメータを使用できます。その他の場合は、`EE_FORCE_ERASE` パラメータを使用する必要があります。

6.3 ワースト·ケース·アクセス時間の短縮

EEPROM の読出しコマンドは、ACTIVE または VALID ページの最高位アドレスから最低位アドレスに向けた Flash 読出しで構成されています。したがって、取得するデータが最初に書き込まれたものだった場合、このプロセスには極めて長時間を要する可能性があります。同様に、ページ転送は、EEPROM エミュレーションで使用されるすべての仮想アドレスを検索し、対応するデータ値を見つけます。これらの読出しおよび転送時間は、エミュレート EEPROM のサイズが大きい場合、極めて長時間になります。

読出しおよび書込みのワースト·ケース·アクセス時間は、要素の値を RAM に保存する LUT（ルックアップ·テーブル）を実装することで短縮できます。LUT を実装してもアクセス時間の代表値には効果がなく、RAM 容量という面でも代償は大きいものの、EEPROM エミュレーション·ドライバの今後のリビジョンにおける実装オプションの一候補としては残るでしょう。

7 まとめ

STM32 デバイスの内部 Flash メモリは EEPROM のエミュレーションに使用でき、多くのアプリケーションにメリットをもたらします。このアプリケーション・ノートでは、メモリ特性は極めて異なるものの、エミュレート EEPROM が次のような面で、本物の EEPORM に匹敵する特性を実現できることを示しました。

- メモリ・サイズ
- 読出しおよび書込みアクセス時間
- ドライバ使用方法のシンプルさと柔軟性
- メモリ書換え耐性
- データ保持期間
- 非同期のリセットや電源障害に対する堅牢性
- 信頼性

そして、最も重要な

- コスト。外付け部品が不要となりコストが削減されます。

8 改版履歴

表 12. 文書改版履歴

日付	版	変更内容
2017 年 7 月 11 日	1	初版発行
2018 年 9 月 20 日	2	STM32L4 および STM32L4+ シリーズ·デバイスを対象製品に含めるためにドキュメントを更新。 更新： – 図 1：ページ・ステータスの遷移 – 図 2：ページ・ステータスの有効な遷移 – セクション 3.3：ページと可変要素のフォーマット – 図 3：Flash のページと EEPROM 可変要素のフォーマット – セクション 4.4：書換えサイクル能力：EEPROM 書換え耐性の向上 – 表 7：4000 バイトのエミュレート EEPROM に対する Flash の使用量（STM32L4/L4+） – セクション 4.7.1：RWW（Read While Write）機能付き Flash メモリを内蔵したデバイス – セクション 5.1.3：ユーザ定義パラメータ – セクション 5.2：EEPROM エミュレーションのメモリ·フットプリント – セクション 5.3：EEPROM エミュレーションのタイミング セクション 4.5：EEPROM エミュレーションに必要な Flash サイズの計算を追加。
2020 年 6 月 1 日	3	更新： – ドキュメントのタイトル – 概要 – セクション 3.3：ページと可変要素のフォーマット および 表 4：Flash メモリのプロパティ – セクション 3.4：簡単な使用例 – セクション 4.4：書換えサイクル能力：EEPROM 書換え耐性の向上および表 6：Flash メモリの書換え耐性 – セクション 4.5：EEPROM エミュレーションに必要な Flash サイズの計算 – セクション 4.6.1：データ·リカバリ – セクション 4.7.3：デュアルコアに関する考慮事項 – セクション 5.1.1：主な特徴 – セクション 5.1.2：STM32Cube 拡張ソフトウェア（X-CUBE-EEPROM）、図 5：ディレクトリ·ツリー、および表 8：対象ボード – セクション 5.1.3：ユーザ定義パラメータ – セクション 5.1.4：ユーザ API の定義および表 9：API 定義 – セクション 5.2：EEPROM エミュレーションのメモリ·フットプリントおよび表 10：EEPROM エミュレーション·メカニズムのメモリ·フットプリント – セクション 5.3：EEPROM エミュレーションのタイミングおよび表 11：各種ターゲットの EEPROM エミュレーションのタイミング。

表 12. 文書改版履歴（続き）

日付	版	変更内容
2020 年 11 月 2 日	4	ドキュメントの対象製品を STM32WL シリーズまで拡張。 更新： – 概要 – 表 1：対象製品 – 表 4：Flash メモリのプロパティ – 表 6：Flash メモリの書換え耐性 – 表 8：対象ボード セクション 4.9：キャッシュ·コヒーレンシのメンテナンスを追加。

表 13. 日本語版文書改版履歴

日付	版	変更内容
2022 年 3 月 15 日	1	日本語版初版発行

重要なお知らせ（よくお読み下さい）

STMicroelectronics NV およびその子会社（以下、ST）は、ST製品及び本書の内容をいつでも予告なく変更、修正、改善、改定及び改良する権利を留保します。購入される方は、発注前にST製品に関する最新の関連情報を必ず入手してください。ST 製品は、注文請書発行時点で有効なST の販売条件に従って販売されます。

ST製品の選択並びに使用については購入される方が全ての責任を負うものとします。購入される方の製品上の操作や設計に関してST は一切の責任を負いません。

明示又は黙示を問わず、STは本書においていかなる知的財産権の実施権も許諾致しません。

本書で説明されている情報とは異なる条件でST 製品が再販された場合、その製品についてSTが与えたいかなる保証も無効となります。

ST およびST ロゴはSTMicroelectronics の商標です。STの登録商標についてはSTウェブサイトをご覧ください。
www.st.com/trademarks
その他の製品またはサービスの名称は、それぞれの所有者に帰属します。

本書の情報は本書の以前のバージョンで提供された全ての情報に優先し、これに代わるものです。

この資料は、STMicroelectronics NV 並びにその子会社(以下ST)が英文で記述した資料（以下、「正規英語版資料」）を、皆様のご理解の一助として頂くためにSTマイクロエレクトロニクス㈱が英文から和文へ翻訳して作成したものです。この資料は現行の正規英語版資料の近時の更新に対応していない場合があります。この資料は、あくまでも正規英語版資料をご理解頂くための補助的参考資料のみにご利用下さい。この資料で説明される製品のご検討及びご採用にあたりましては、必ず最新の正規英語版資料を事前にご確認下さい。ST及びSTマイクロエレクトロニクス㈱は、現行の正規英語版資料の更新により製品に関する最新の情報を提供しているにも関わらず、当該英語版資料に対応した更新がなされていないこの資料の情報に基づいて発生した問題や障害などにつきましては如何なる責任も負いません。