本ドキュメントは Cypress (サイプレス) 製品に関する情報が記載されております。本ドキュメントには、「MB」から始まるシリーズ名，品名およびオーダ型格が記載されておりますが、これらはすべて「CY」から始まるシリーズ名，品名およびオーダ型格として、新規および既存のお客様に引き続き提供してまいります。

オーダ型格の調べ方について

1. www.cypress.com/pcnにアクセスしてください。
2. SEARCH PCNS フィールドに、オーダ型格などのキーワードを入力し、「Apply」をクリックしてください。
3. 該当するタイトル(Title)をクリックしてください。
4. 「Affected Parts List」ファイルを開いてください。
   当該ファイルに記載されている各種変更情報をご利用ください。

詳しいお問い合わせ先

Cypress 製品およびそのソリューションの詳細につきましては、お近くの営業所へお問い合わせください。

サイプレスについて

サイプレスは、世界で最も革新的な車載や産業機器、スマート家電、民生機器および医療機器製品向けに、最先端の組み込みシステム ソリューションを提供するリーディング カンパニーです。サイプレスのマイクロコントローラーや、アナログ IC、ワイヤレスおよび USB ベースのコネクティビティ ソリューション、高い信頼性と高性能を提供するメモリ製品は、各種機器メーカーの差異化製品の開発と早期市場参入を支援します。サイプレスは、ベストクラスのサポートと開発リソースをグローバルに提供することで、彼らが従来市場を破壊しまったく新しい製品カテゴリを歴史的なスピードで市場投入できるよう支援します。詳細はサイプレスのウェブサイト (japan.cypress.com) をご覧ください。

AN204370

8FX ファミリ スターターキットを用いたフラッシュオペレーション例

関連ファミリ：	シリーズ名	品種型格
	MB95560H	MB95F564H/K, MB95F563H/K, MB95F562K/H
	MB95570H	MB95F574H/K, MB95F573H/K, MB95F572K/H
	MB95580H	MB95F584H/K, MB95F583H/K, MB95F582H/K

このアプリケーションノートでは, Cypress New 8FX ファミリ MB95560H/570H/580H シリーズを例としたフラッシュの書換え処理について説明します。

Contents

1 はじめに 1
2 プログラミングのアルゴリズム 1
2.1 概要 1
2.2 フラッシュメモリの特長 1
2.3 セクタ構成 2
2.4 フラッシュメモリのレジスタ 2
2.5 フラッシュメモリの自動アルゴリズム起動 3
3 ソースコード 3
3.1 flash.asm 3
3.2 Main.c 7
4 フラッシュ処理に関する注意事項 10
5 改訂履歴 11
セールス, ソリューションおよび法律情報 12

1 はじめに

このアプリケーションノートでは, Cypress New 8FX ファミリ MB95560H/570H/580H シリーズを例としたフラッシュの書換え処理について説明します。他の New 8FX ファミリ共通で応用が可能です。プログラミングのアルゴリズム説明, flash.asm, フラッシュ消去の C ソースコード, フラッシュ書き込みの C ソースコード, および Main.c を含む包括的なプロジェクトについて説明します。

2 プログラミングのアルゴリズム

フラッシュ処理のアルゴリズム

2.1 概要

本品種に搭載されているデュアルオペレーションフラッシュメモリはユーザプログラムがフラッシュメモリ上で動作している場合でも, バンクの異なるフラッシュメモリの消去や, 書込みが可能です。アプリケーションノートではフラッシュメモリの上位バンクより下位バンクに対するセクタ書込み, セクタ消去の例を示します。

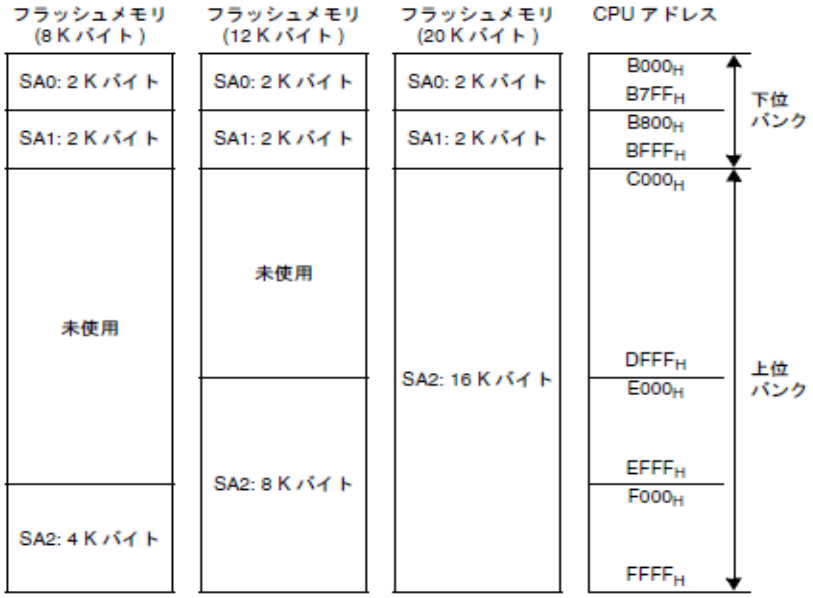
2.2 フラッシュメモリの特長

- セクタ構成: 上位バンク 4KB / 8KB / 16KB + 下位バンク 2KB x2
- 自動書込みアルゴリズム (Embedded Algorithm)
- データポーリングによる書込み / 消去完了の検出
- CPU 割込みによる書込み / 消去完了の検出
- JEDEC 標準規格コマンドとの互換性
- 消去 / 書込みサイクル (最小): 100,000 回

2.3 セクタ構成

図 1 は, 8/12/20 KB フラッシュメモリのセクタ構成を示しています。図には, 各セクタの上位および下位アドレスが示されています。

図 1. 8/12/20 KB フラッシュメモリのセクタ構成



2.4 フラッシュメモリのレジスタ

図 2 に, フラッシュメモリのレジスタを示します。詳細については, MB95560H/570H/580H シリーズのハードウェアマニュアル第 20 章を参照してください。

図 2. 8/12/20 KB フラッシュメモリのセクタ構成

フラッシュメモリステータスレジスタ 2 (FSR2)

アドレス	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	初期値
0071_H	PEIEN	PGMEND	PTIEN	PGMTO	EEIEN	ERSEND	ETIEN	ERSTO	00000000_B
	R/W	R(RM1),W	R/W	R(RM1),W	R/W	R(RM1),W	R/W	R(RM1),W	

フラッシュメモリステータスレジスタ (FSR)

アドレス	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	初期値
0072_H	-	-	RDYIRQ	RDY	予約	IRQEN	WRE	SSEN	$000X0000_B$
	R0/WX	R0/WX	R(RM1),W	R/WX	R/W0	R/W	R/W	R/W	

フラッシュメモリセクタ書込み制御レジスタ 0 (SWRE0)

アドレス	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	初期値
0073_H	予約	予約	予約	予約	予約	SA2E	SA1E	SA0E	00000000_B
	R/W0	R/W0	R/W0	R/W0	R/W0	R/W	R/W	R/W	

フラッシュメモリステータスレジスタ 3 (FSR3)

アドレス	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	初期値
0074_H	-	-	-	CERS	ESPS	SERS	PGMS	HANG	$000XXXXX_B$
	R0/WX	R0/WX	R0/WX	R/WX	R/WX	R/WX	R/WX	R/WX	

フラッシュメモリステータスレジスタ 4 (FSR4)

アドレス	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	初期値
0075_H	-	CEREND	CTIEN	CERTO	-	-	-	-	00000000_B
	R0/WX	R(RM1),W	R/W	R(RM1),W	R0/WX	R0/WX	R0/WX	R0/WX	

2.5 フラッシュメモリの自動アルゴリズム起動

フラッシュメモリの自動アルゴリズムを呼び出す基本コマンドには, 読出し / リセット, 書込み, チップ消去, セクタ消去という 4 つのタイプがあります。図 3 にコマンドの一覧を示します。

図 3. コマンドシーケンス

コマンドシーケンス	バス書込みサイクル	最初のバス書込みサイクル		2 回目のバス書込みサイクル		3 回目のバス書込みサイクル		4 回目のバス書込みサイクル		5 回目のバス書込みサイクル		6 回目のバス書込みサイクル	
		アドレス	データ	アドレス	データ	アドレス	データ	アドレス	データ	アドレス	データ	アドレス	データ
読出し / リセット	1	0xUXXX	0xF0	-	-	-	-	-	-	-	-	-	-
書込み	4	0xUAAA	0xAA	0xU554	0x55	0xUAAA	0xA0	PA	PD	-	-	-	-
チップ消去	6	0xUAAA	0xAA	0xU554	0x55	0xUAAA	0x80	0xUAAA	0xAA	0xU554	0x55	0xUAAA	0x10
セクタ消去	6	0xUAAA	0xAA	0xU554	0x55	0xUAAA	0x80	0xUAAA	0xAA	0xU554	0x55	SA	0x30
アンロックバイパスエントリ	3	0xUAAA	0xAA	0xU554	0x55	0xUAAA	0x20	-	-	-	-	-	-
アンロックバイパス書込み	2	0xUXXX	0xA0	PA	PD	-	-	-	-	-	-	-	-
アンロックバイパスリセット	2	0xUXXX	0x90	0xUXXX	any	-	-	-	-	-	-	-	-
セクタ消去一時停止		アドレス "0xUXXX" にデータ "0xB0" を入力することによって, セクタ消去を一時停止します。											
セクタ消去再開		アドレス "0xUXXX" にデータ "0x30" を入力することによって, セクタ消去一時停止後, 消去を再開します。											
消去セクタ追加		SA にデータ "0x30" を入力することによって, 新しい消去セクタを追加します。											

PA : 書込みアドレス
SA : セクタアドレス (セクタ内の任意のアドレスを指定する。)
PD : 書込みデータ
U : 上位 4 ビットは, 書込みが許可されているセクタ内の任意のアドレスです。
X : 任意の値
any : 任意の書込みデータ

MB95560H シリーズのハードウェアマニュアル記載のコマンドアドレスに 0xUAA8 と記載のある版数のものがあります。他の New 8FX ファミリ同様に 0xUAAA にてアクセスが可能です。本アプリケーションノートではコマンドアドレスを 0xUAAA として取り扱っています。

3 ソースコード

この章では、サンプルコードについて説明します。

3.1 flash.asm

flash.asm は、フラッシュ書込み, セクタ消去を実行するためのアセンブリコードです。

_EraseStart は, flash.asm 内のセクタフラッシュ消去ルーチンの開始アドレスです。

_WriteStart は, flash.asm 内のフラッシュ書込みルーチンの開始アドレスです。

それらは, main () 内の命令によってコールできるようにエクスポートしています。

```
.EXPORT _EraseStart
.EXPORT _WriteStart
```

SWRE0 にて書込みを行うセクタの許可を行います。

セクタ SA0 または SA1 への消去書込みを行うには SA0E/SA1E の両方を ”1” に設定します。

```
MOV    A, #03H      ; SA0E/SA1E write enable
MOV    SWRE0,A
```

消去アドレスを EP から push して保存します。アドレスは main ()で割り当てられます。

```
_EraseStart:
                          ;
      MOVW   A,EP
      PUSHW  A            ; Save erase address
```

書込みアドレスを EP から push して保存します。書き込みデータを IX から push して保存します。これらは main () で割り当てられます。

```
_WriteStart:
                    ;
_write_address:
      PUSHW  IX           ; Save write data
      MOVW   A,EP
      PUSHW  A            ; Save write address
```

フラッシュの自動アルゴリズムの消去および書込みに基づいて, UAAAH と U554H の U は, 書込みアドレスの上位 4 ビットと同じです。次のコードは, U を計算にて正しい UAAAH と U554H を取得します。

```
      MOVW   A,#0xF000
      ANDW   A
      MOVW   A,#0x0AAA
      ORW    A
      MOVW   EP,A         ; 0x0AAA | (address & 0xf000)
      XCHW   A,T
      MOVW   A,#0x0554
      ORW    A
      MOVW   IX,A         ; 0x0554 | (address & 0xf000)
```

レジスタ FSR でビット WRE を設定することによって, フラッシュ書込みを有効にします。

```
SETB   FSR:WRE      ; Write Enable
```

フラッシュメモリ セクタ消去の自動アルゴリズム

```
MOV    A,#0xAA      ; 0x*AAAH <= 0xAA
MOV    @EP,A

MOV    A,#0x55      ; 0x*554 <= 0x55
MOV    @IX,A
MOV    A,#0x80      ; 0x*AAAH <= 0x80
MOV    @EP,A

MOV    A,#0xAA      ; 0x*AAAH <= 0xAA
MOV    @EP,A

MOV    A,#0x55      ; 0x*554 <= 0x55
MOV    @IX,A

POPW   A            ; Restore Erase address SA
MOVW   EP,A
MOV    A,#30H       ; The last data.  Sector erase
MOV    @EP,A        ; Start Erase
```

フラッシュメモリ書込みの自動アルゴリズム

```
MOV    A,#0xAA       ; 0xUAAAH <= 0xAA
MOV    @EP,A

MOV    A,#0x55       ; 0xU554 <= 0x55
MOV    @IX,A

MOV    A,#0xA0       ; 0xUAAAH <= 0xA0
MOV    @EP,A

POPW   A
MOVW   EP,A          ; write address PA
POPW   IX

MOVW   A,IX          ; write data PD
MOV    @EP,A         ; to write flash
```

消去ループ

消去/書込み消去コマンド発行後、RDY ビットが ”0” となるまでに 2 マシンクロック (MCLK) の遅延があります。そのため RDY ビット確認処理ループ実行前に NOP を 2 回挿入します。詳細はハードウェアマニュアルの第 20 章を参照ください。

```
      NOP
      NOP
EraseLoop:
      BBS   FSR:RDY,EraseEnd ; Erase Flash successes?

      MOV   A,@EP
      AND   A,#0x20          ; Check Time Out?
      BZ    EraseLoop

      BBS   FSR:RDY,EraseEnd ; Erase Flash successes?
      NOP
      BBS   FSR:RDY,EraseEnd ; Erase Flash successes?
```

書込みループ

```
      NOP
      NOP
WriteLoop:
      BBS   FSR:RDY,WriteEnd ; write Flash successes?

      MOV   A,@EP
      AND   A,#0x20          ; to check time out?
      BZ    WriteLoop

      BBS   FSR:RDY,WriteEnd ; write Flash successes?
      NOP
      BBS   FSR:RDY,WriteEnd ; write Flash successes?
```

フラッシュ消去失敗, フラッシュのリセット, および A レジスタでのエラーフラグ設定

```
EraseError:
      MOV   A,#0xF0
      MOV   0xFF00,A     ; Reset Flash
      MOV   A,#01H       ; Set error Flag
```

フラッシュ書込みの失敗, フラッシュのリセット, および A レジスタでのエラーフラグ設定

```
WriteError:
      MOV   A,#0xF0
      MOV   0xFF00,A    ; Reset Flash
      MOV   A,#01H            ; Set error Flag
```

レジスタ FSR でビット WRE をクリアすることによって, フラッシュ書込みを無効にします。

```
      CLRB   FSR:WRE     ; write disable
```

フラッシュ消去が成功した場合は, 成功フラグを A レジスタへ設定します。

```
EraseEnd:
      MOV   A,#00H                        ; normal ack
      MOVW  EP,A
```

フラッシュ書込みが成功した場合は, 成功フラグを A レジスタへ設定します。

```
WriteEnd:
      MOV   A,#00H                        ; normal ack
      MOVW  EP,A
```

3.2 Main.c

Main.c は, フラッシュ書込み, セクタ消去を実行するための C コードです。

クローバル変数を定義します。

```
unsigned char  result, Flag;
unsigned short address;
unsigned char  data;
```

3.2.1 消去アドレスの指定

アセンブリおよび C 言語のコンパイル規則に基づくと，アセンブリ言語の変数「_address」は，C 言語の変数「address」と同じです。次のコードは，main () で割り当てられた消去アドレスを EP に転送します。これは，flash.asm で使用されます。

```
MOVW A, _address
MOVW EP,A
```

アセンブリフラッシュ消去ルーチンは，次のコードでコールされます。

```
CALL _EraseStart
```

_EraseStart は，flash.asm 内のフラッシュ消去ルーチンの開始アドレスです。

3.2.2 書込みアドレス指定

次のコードは，main () で割り当てられた書込みアドレスを EP に転送し，データを flash.asm で使用される IX に転送します。

```
MOV  A,_data      ;write data
MOVW IX,A
MOVW A, _address  ;write address
MOVW EP,A
```

アセンブリフラッシュ書込みルーチンは，次のコードでコールされます。

```
CALL _WriteStart
```

_WriteStart は，flash.asm 内のフラッシュ書き込みルーチンの開始アドレスです。

3.2.3 プログラミング機能の使用方法

次に，サンプルの C ソースメインコードを示します。Main.c からの flash.asm の使用方法を示しています。コード全体についてはサンプルコードを参照してください。

EraseStart は，flash.asm 内のフラッシュ消去ルーチンの開始アドレスです。WriteStart は，flash.asm 内のフラッシュ書込みルーチンの開始アドレスです。EraseStart と WriteStart は，main () 内の命令によってコールできるようにインポートされています。

アセンブリおよび C 言語のコンパイル規則に基づき，アセンブリ言語の「_EraseStart」は C 言語の「EraseStart」と同じ，「_WriteStart」は「WriteStart」と同じです。

```
extern EraseStart;
extern WriteStart;
```

ユーザは，フラッシュ処理の結果に応じた，エラー発生時の処理や成功時の処理を追加できます。サンプルではポートへの出力を行っています。

```
void error(void)
{
      IO_PDR0.bit.P05=0;            // do something here LED2
}
void success(void)
{
       IO_PDR6.bit.P64=0;           // do something here LED3
}
```

消去/書込みアドレス「0xB000」(SA0) および書込みデータ「0xA0」は，グローバル変数のアドレスとデータによって flash.asm に転送されます。書込みアドレス「0xB000」(SA0) とデータ「0xA0」は，ユーザによって割り当てを行います。サンプルではアドレスとデータを固定しています。

3.2.4 セクタ消去処理の呼び出し

flash_erase () がコールされ, セクタ消去の結果がフラグに返ります。

```
//-------------------------------------------------------
// Sector Erase flash
//-------------------------------------------------------
      address = 0xB000;             // set write address (SA0)
      Flag = flash_erase();         // flash sector erase routine
```

3.2.5 書込み処理の呼び出し

flash_write () がコールされ, フラッシュ書き込みの結果がフラグに返ります。

```
//-------------------------------------------------------
// Write
//-------------------------------------------------------
      address = 0xB000;       // set write address (SA0)
      data = 0xA0;            // set write data
      Flag = flash_write();   // flash write routine
```

3.2.6 フラグの判定

Flag ==1 はフラッシュ書込み失敗であり, error () で何らかの処理を行います。

Flag == 0 はフラッシュ書込み成功であり, success () で何らかの処理を行います。

```
if (Flag == 1)
      error();
else
      success();
```

4 フラッシュ処理に関する注意事項

この項では, フラッシュ処理に関する注意事項を示します。

サンプルではフラッシュのセクタ消去コマンドを発行しています。チップ消去コマンドの処理を行った場合, NVR 領域のデータはすべて消去され, 内蔵 CR トリミングのプリセット値などが消去されてしまいますのでご注意ください。

詳しくはハードウェアマニュアルの NVR の章を参照してください。

5 改訂履歴

文書名: AN204370 - 8FX ファミリ スターターキットを用いたフラッシュオペレーション例

文書番号: 002-04331

版	ECN 番号	変更者	発行日	変更内容
**	-	HUAL	03/05/2013	新規作成
			01/31/2014	社名変更および記述フォーマットの変換
*A	5764715	HUAL	06/06/2017	これは英語版の 002-04370 Rev. *A を翻訳した日本語版です。

セールス, ソリューションおよび法律情報

ワールドワイドな販売と設計サポート

サイプレスは、事業所、ソリューション センター、メーカー代理店、および販売代理店の世界的なネットワークを保持しています。お客様の最寄りのオフィスについては、サイプレスのロケーション ページをご覧ください。

製品

ARM® Cortex® Microcontrollers	cypress.com/arm
車載用	cypress.com/automotive
クロック&バッファ	cypress.com/clocks
インターフェース	cypress.com/interface
IoT (モノのインターネット)	cypress.com/iot
メモリ	cypress.com/memory
マイクロコントローラ	cypress.com/mcu
PSoC	cypress.com/psoc
電源用 IC	cypress.com/pmic
タッチ センシング	cypress.com/touch
USB コントローラー	cypress.com/usb
ワイヤレス／RF	cypress.com/wireless

PSoC® ソリューション

PSoC 1 | PSoC 3 | PSoC 4 | PSoC 5LP | PSoC 6

サイプレス開発者コミュニティ

フォーラム | WICED IOT Forums | Projects | ビデオ | ブログ | トレーニング | Components

テクニカルサポート

cypress.com/support

All other trademarks or registered trademarks referenced herein are the property of their respective owners.

Cypress Semiconductor
198 Champion Court
San Jose, CA 95134-1709

© Cypress Semiconductor Corporation, 2013-2017. 本書面は、Cypress Semiconductor Corporation 及び Spansion LLC を含むその子会社（以下、「Cypress」という。）に帰属する財産である。本書面（本書面に含まれ又は言及されているあらゆるソフトウェア又はファームウェア（以下、「本ソフトウェア」という。）を含む）は、アメリカ合衆国及び世界のその他の国における知的財産法令及び条約に基づき、Cypress が所有する。Cypress はこれらの法令及び条約に基づく全ての権利を留保し、また、本段落で特に記載されているものを除き、Cypress の特許権、著作権、商標権又はその他の知的財産権のライセンスを一切許諾していない。本ソフトウェアにライセンス契約書が伴っておらず、かつ、あなたが Cypress との間で別途本ソフトウェアの使用方法を定める書面による合意をしていない場合、Cypress は、あなたに対して、（1）本ソフトウェアの著作権に基づき、（a）ソースコード形式で提供されている本ソフトウェアについて、Cypress ハードウェア製品と共に用いるためにのみ、組織内部でのみ、本ソフトウェアの修正及び複製を行うこと、並びに（b）Cypress のハードウェア製品ユニットに用いるためにのみ、（直接又は再販売者及び販売代理店を介して間接のいずれかで）エンドユーザーに対して、バイナリーコード形式で本ソフトウェアを外部に配布すること、並びに（2）本ソフトウェア（Cypress により提供され、修正がなされていないもの）に抵触する Cypress の特許権のクレームに基づき、Cypress ハードウェア製品と共に用いるためにのみ、本ソフトウェアの作成、利用、配布及び輸入を行うことについての非独占的で譲渡不能な一身専属的ライセンス（サブライセンスの権利を除く）を付与する。本ソフトウェアのその他の使用、複製、修正、変換又はコンパイルを禁止する。

適用される法律により許される範囲内で、Cypress は、本書面又はいかなる本ソフトウェアに関しても、明示又は黙示をとわず、いかなる保証（商品性及び特定の目的への適合性の黙示の保証を含むがこれらに限られない）も行わない。適用される法律により許される範囲内で、Cypress は、別途通知することなく、本書面を変更する権利を留保する。Cypress は、本書面に記載のあるいかなる製品又は回路の適用又は使用から生じる一切の責任を負わない。本書面で提供されたあらゆる情報（あらゆるサンプルデザイン情報又はプログラムコードを含む）は、参照目的のためのみに提供されたものである。この情報で構成するあらゆるアプリケーション及びその結果としてのあらゆる製品の機能性及び安全性を適切に設計し、プログラムし、かつテストすることは、本書面のユーザーの責任において行われるものとする。Cypress 製品は、兵器、兵器システム、原子力施設、生命維持装置若しくは生命維持システム、蘇生用の設備及び外科的移植を含むその他の医療機器若しくは医療システム、汚染管理若しくは有害物質管理の運用のために設計され若しくは意図されたシステムの重要な構成部分として用いるため、又はシステムの不具合が人身傷害、死亡若しくは物的損害を生じさせることになるその他の使用（以下、「本目的外使用」という。）のためには、設計、意図又は承認されていない。重要な構成部分とは、装置又はシステムのその構成部分の不具合が、その装置若しくはシステムの不具合を生じさせるか又はその安全性若しくは実効性に影響すると合理的に予想できる、機器又はシステムのあらゆる構成部分をいう。Cypress 製品のあらゆる本目的外使用から生じ、若しくは本目的外使用に関連するいかなる請求、損害又はその他の責任についても、Cypress はその全部又は一部をとわず一切の責任を負わず、かつ、あなたは Cypress をそれら一切から免除するものとし、本書により免除する。あなたは、Cypress 製品の本目的外使用から生じ又は本目的外使用に関連するあらゆる請求、費用、損害及びその他の責任（人身傷害又は死亡に基づく請求を含む）から Cypress を免責補償する。

Cypress、Cypress のロゴ、Spansion、Spansion のロゴ及びこれらの組み合わせ、WICED、PSoC、CapsSense、EZ-USB、F-RAM、及び Traveo は、米国及びその他の国における Cypress の商標又は登録商標である。Cypress の商標のより完全なリストは、cypress.com を参照のこと。その他の名称及びブランドは、それぞれの権利者の財産として権利主張がなされている可能性がある。