ご注意：この日本語版ドキュメントは、参考資料としてご使用の上、最新情報につきましては、必ず英語版オリジナルをご参照いただきますようお願いします。

セクション 23. CodeGuard™ セキュリティ

ハイライト

本セクションには以下の主要項目を記載しています。

23.1 コード保護の概要

マイクロチップ社の CodeGuard™ セキュリティを使用すると、1 つのチップ上で複数の企業がリソース (メモリ、割り込み、周辺モジュール) を安全に共有できます。この内蔵コード保護機能を利用する事で、知的財産 (IP) ベンダ、オリジナル設計 / オリジナル装置メーカー (ODM/OEM)、付加価値を追加するリセラー (VAR) にとって以下の利点があります。

- システムコストの削減
- 部品点数の削減とそれに伴う在庫管理の容易化
- 未認定パートナー企業への IP 流出リスクの低減
- コードの配布とフラッシュメモリ更新時におけるセキュリティの強化

dsPIC33F の内蔵プログラム フラッシュ メモリのコード空間は 3 種類のセグメントによる構成が可能です。これらのセグメントには、それぞれにセキュリティ権限レベルとシステム機能が暗黙のうちに設定されています。

1. ブートセグメント (BS) は、最高レベルのセキュリティ権限を持ちます。他のセグメントに対するより高いアクセス権を持ちます。ブートセグメントの目的はブートローダとデバイス更新機能を保護する事です。
2. セキュア セグメント (SS) は、ブートセグメントに次ぐセキュリティ権限を持ちます。このセグメントの目的はアルゴリズム ベンダの独自アルゴリズムを格納する事です。
3. 汎用セグメント (GS) は、最も低いセキュリティ権限を持ちます。エンドユーザのシステムコード向けです。

デバイスのユーザデータ RAM 空間の一部をセキュア RAM として割り当てる事もできます。これらのセグメントはブートセグメントまたはセキュア セグメントに直接関連付けられます。

コードまたはデータ内容を暴露する恐れのある全てのシステム動作は、その起動元セグメントまたは動作の対象であるセグメントが持つセキュリティ権限に応じて制限されます。

制限を受ける動作には以下のようなものがあります。

- プログラミング、消去、ベリファイ動作
- コード空間の読み出し、書き込み
- 保護データ空間の読み出し、書き込み
- セグメント外からセキュア セグメント内へのコードフロー変更
- セキュア セグメント内への割り込みベクタ

セキュア セグメントとそのパラメータへアクセスするには、コンフィグレーション ビットを使います。これらのビットにより、プログラム フラッシュ メモリおよび RAM セグメントの容量と制限の両方を設定できます。

23.2 デバイス固有のコード保護機能

dsPIC33F は、CodeGuard セキュリティ機能の 2 種類のサブセットを提供します。

メモリ容量の小さいデバイスでは、プログラムメモリをブートセグメントと汎用セグメントに割り当てる事ができますが、セキュア セグメントおよびデータ RAM 保護機能はありません。

メモリ容量の大きいデバイスの場合、メモリをブートセグメント、セキュア セグメント、汎用セグメントのコード空間として割り当てる事ができます。さらに、データ RAM をブートセグメントとセキュア セグメントに割り当てる事も可能です。

表 23-1 に使用可能なコード保護機能を示します。これらの機能と特定のデバイスまたは製品ファミリとの対応は、各デバイスのデータシートを参照してください。

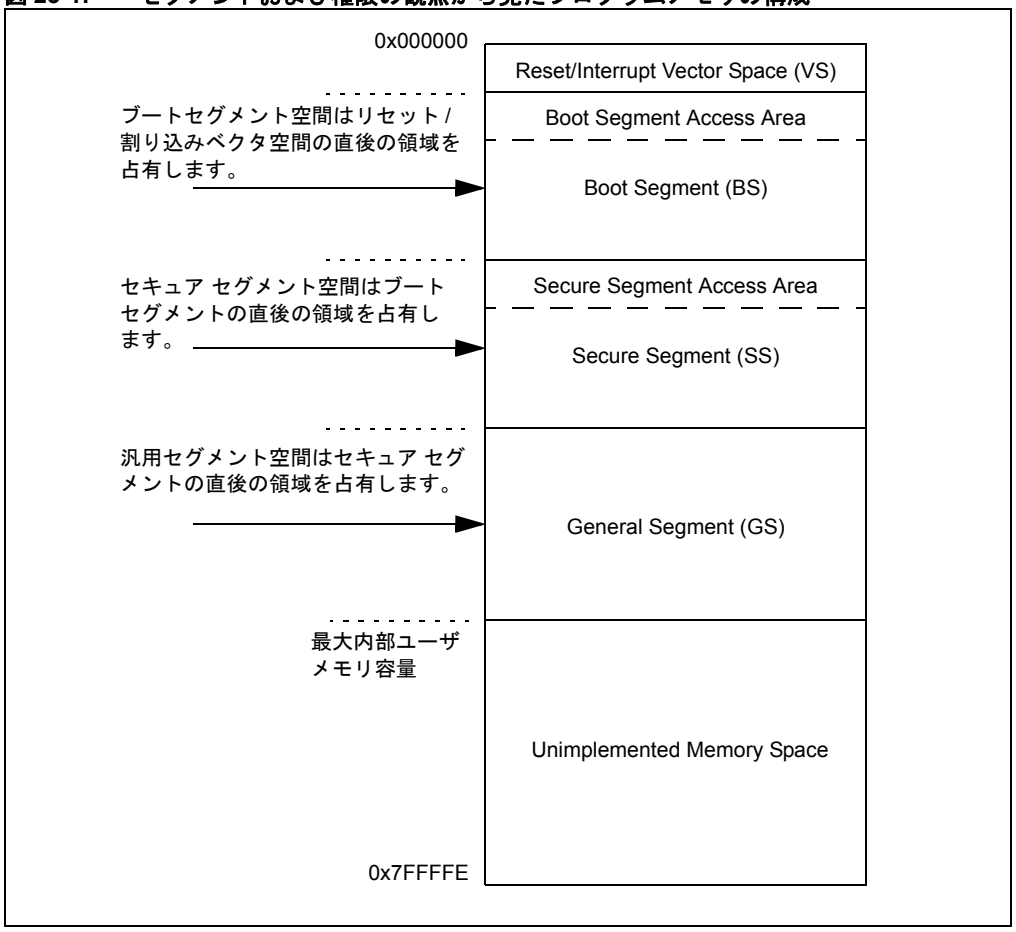
表 23-1: コード保護機能

機能	メモリ容量の小さいデバイス	メモリ容量の大きいデバイス
コード空間のブートセグメントとしての割り当て	可	可
コード空間のセキュア セグメントとしての割り当て	—	可
コード空間の汎用セグメントとしての割り当て	可	可
データ RAM のブートおよびセキュア セグメントへの割り当て	—	可

23.3 プログラムメモリの構成

ユーザ プログラムメモリ全体を 3 種類のセグメントのいずれか 1 つに割り当てる事ができます。これらセグメントの容量は、コンフィグレーション ビットによって決まります。セグメント間の相対位置は不変です。例えば、ブートセグメントを割り当てた場合、そのメモリ領域はデバイス割り込みベクタ空間の直後を占有します。セキュア セグメントが存在する場合はブートセグメントの直後に置かれ、汎用セグメントはさらにその直後の領域を占有します (図 23-1 参照)。

図 23-1: セグメントおよび権限の観点から見たプログラムメモリの構成

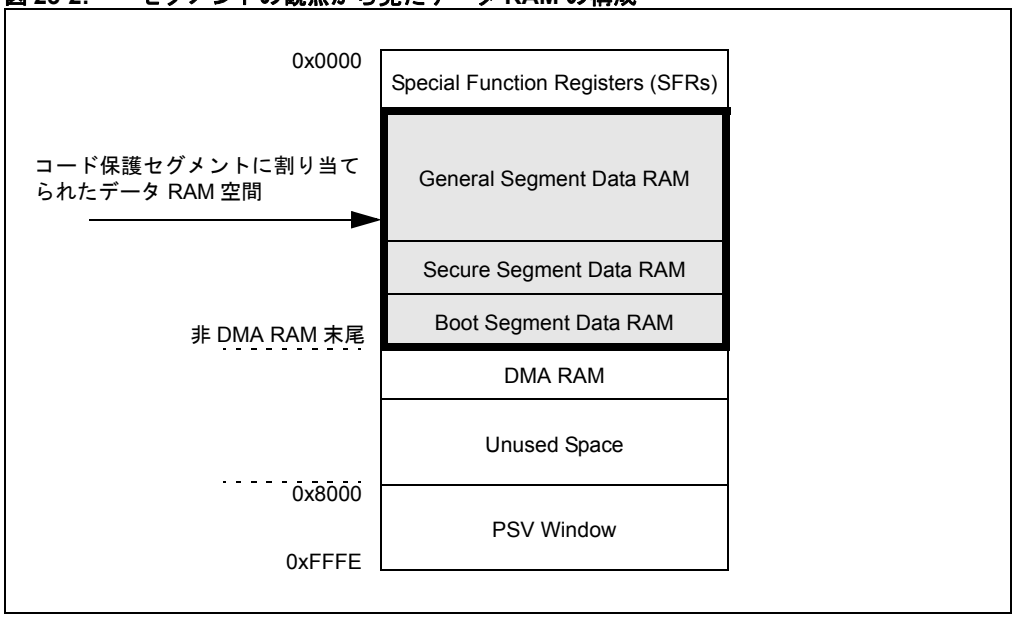


23.4 データ RAM の構成

データ RAM メモリも、ブート、セキュア、汎用の各コード保護セグメントに割り当てる事ができます。セグメントの容量は主にコンフィグレーション ビットによって指定します。セグメント間の相対位置は不変です。つまり、図 23-2 に示す通り、ブートセグメント RAM 領域は非 DMA RAM 末尾のメモリ領域を占有し、セキュア セグメント RAM はブートセグメントの直前、汎用セグメント RAM は残りのデータ RAM 空間を占有します。

Note:	DMA RAM は全ての dsPIC33F に実装されているわけではありません。DMA RAM の有無と容量については各デバイスのデータシートを参照してください。

図 23-2: セグメントの観点から見たデータ RAM の構成



23.5 制御レジスタ

複数のコンフィグレーションおよび特殊機能レジスタ (SFR) がセキュリティ機能を制御します。基本および中程度のセキュリティ実装では、これらのレジスタの一部が存在しない場合があります。コード セキュリティ機能をサポートする主要レジスタは以下の通りです。

- **FBS: ブートセグメント コンフィグレーション レジスタ バイト**
- **BSRAM: ブートセグメント RAM 特殊機能レジスタ**
- **FSS: セキュア セグメント コンフィグレーション レジスタ バイト**
- **SSRAM: セキュア セグメント RAM 特殊機能レジスタ**
- **FGS: 汎用セグメント コンフィグレーション レジスタ**
- **INTTREG: 割り込みベクタおよび優先度ステータス特殊機能レジスタ**

23.6 ブートセグメント (BS)

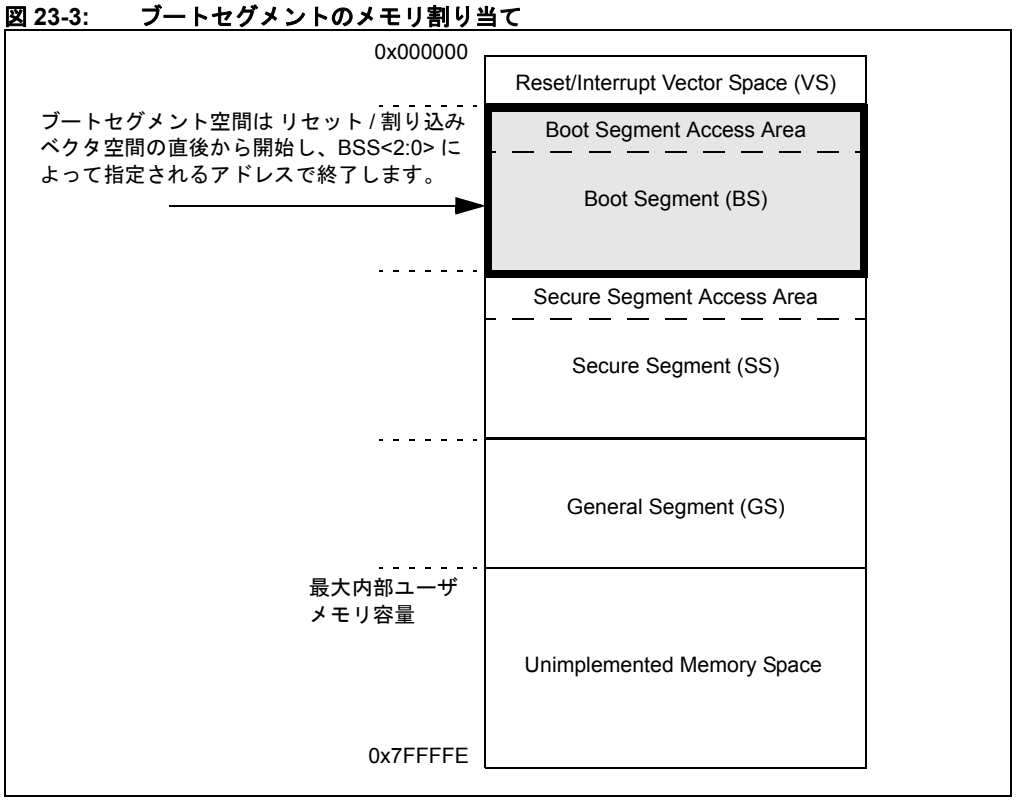
ブートセグメントは、最高レベルのセキュリティ権限を持ちます。このセグメントには、容量を小さく設定して簡素でセキュアなブートローダを格納するか、より大きな容量を割り当てて、高度でセキュアなオペレーティング システムを格納する事ができます。

ブートセグメントは自身の領域内のデータを書き換える事ができるため、「暗号化キー」等のデータの保存および更新が可能です。

23.6.1 ブートセグメントの割り当て

ブートセグメントの有無と容量は、コンフィグレーション ビット BSS<2:0> (FBS<3:1>) によって決まります。消去状態で未プログラムのデバイスの既定オプションでは、ブートセグメントは割り当てられません。ブートセグメントを実装した場合、その開始位置は割り込みベクタ空間の終端となり、BSS<2:0> ビットによって指定されるアドレスで終了します。

図 23-3: ブートセグメントのメモリ割り当て



23.6.1.1 ブートセグメントの容量オプション

表 23-2 にブートセグメントの容量オプションの例を示します (64 KB フラッシュメモリ搭載デバイスの場合)。ここに示したプログラムメモリの開始アドレスと終了アドレスは代表例です。デバイス固有のプログラムメモリ アドレスは、各デバイスのデータシートを参照してください。

表 23-2: ブートセグメント容量の例

BSS2:BSS0	セキュリティ レベル	BS 容量	BS 開始 アドレス	BS 終了 アドレス
x11	ブート プログラム フラッシュ セグメントなし			
110	標準	小	0x000200	0x0007FE
010	高	小	0x000200	0x0007FE
101	標準	中	0x000200	0x001FFE
001	高	中	0x000200	0x001FFE
100	標準	大	0x000200	0x003FFE
000	高	大	0x000200	0x003FFE

レジスタ 23-1: FBS: ブートセグメント コンフィグレーション レジスタ バイト

下位第 3 バイト :

R/P	R/P	r	r	R/P	R/P	R/P	R/P
RBS<1:0>(1)		—	—	BSS<2:0>(2)			BWRP(3)
bit 7							bit 0

凡例 :	r = 予約	P = プログラム可能ビット
R = 読み出し可能ビット	W = 書き込み可能ビット	U = 未実装ビット、「0」として読み出し
-n = POR 時の値	「1」= ビットをセット	「0」= ビットをクリア x = ビットは未知

bit 7-6 **RBS<1:0>:** ブートセグメント RAM コード保護ビット (1)
11 = ブート RAM を定義しない
10 = 128 バイトのブート RAM を定義する
01 = 256 バイトのブート RAM を定義する
00 = 1024 バイトのブート RAM を定義する

bit 5-4 **予約 :** 使用不可です。プログラミング値は無効になります

bit 3-1 **BSS<2:0>:** ブートセグメント プログラム フラッシュ コード保護ビット (2)
x11 = ブート プログラム フラッシュ セグメントを定義しない
110 = 標準セキュリティ、小容量ブートセグメント
010 = 高セキュリティ、小容量ブートセグメント
101 = 標準セキュリティ、中容量ブートセグメント
001 = 高セキュリティ、中容量ブートセグメント
100 = 標準セキュリティ、大容量ブートセグメント
000 = 高セキュリティ、大容量ブートセグメント

bit 0 **BWRP:** ブートセグメント プログラム フラッシュ書き込み保護ビット (3)
1 = ブートセグメントへの書き込み許可
0 = ブートセグメントへの書き込み禁止

Note 1: ブートセグメント RAM コード保護機能は全てのデバイスに搭載されているわけではありません。各デバイス固有の情報は、表 23-3、表 23-4、表 23-5 を参照してください。

2: ブートセグメント容量の大、中、小の厳密な定義はデバイスごとに異なります。各デバイス固有の情報は、表 23-6、表 23-7、表 23-8 を参照してください。

3: ブートセグメントが不要の場合、BWRP ビットを「1」にプログラムする必要があります。

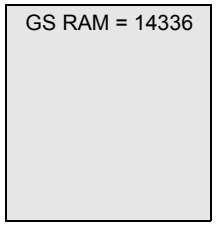
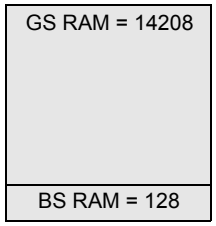
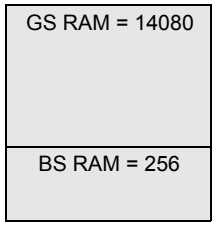
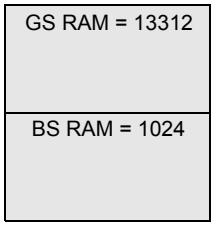
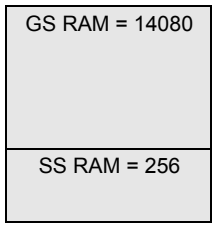
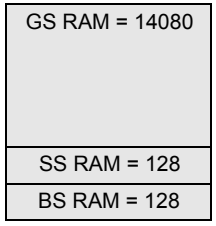
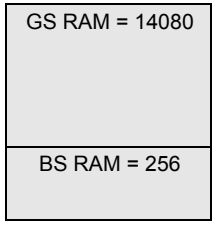
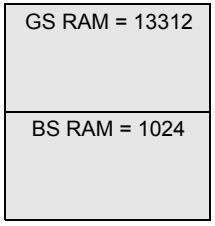
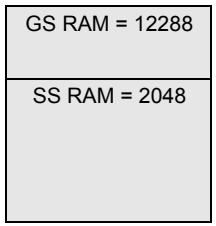
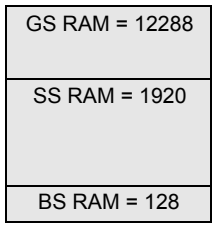
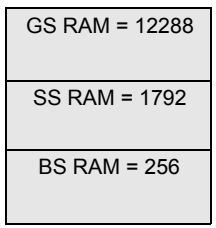
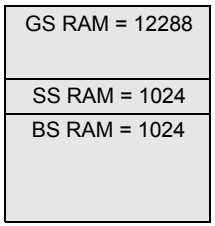
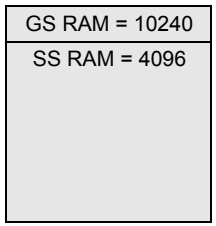
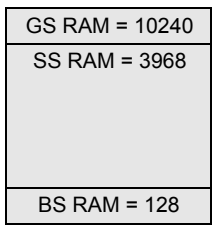
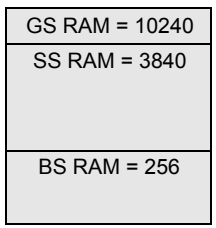
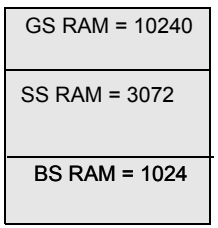
表 23-3: 30 KB RAM を搭載したデバイスのデータ RAM のセグメント容量

コンフィグレーション ビット	RBS<1:0> = 11 OR RBS<1:0> = 10 AND RL_BSR = 1	RBS<1:0> = 10 AND RL_BSR = 0 OR RBS<1:0> = 01 AND RL_BSR = 1	RBS<1:0> = 01 AND RL_BSR = 0 OR RBS<1:0> = 00 AND RL_BSR = 1	RBS<1:0> = 00 AND RL_BSR = 0
RSS<1:0> = 11 OR RSS<1:0> = 10 AND RL_SSR = 1	GS RAM = 28672 0x0800 0x77FF	GS RAM = 28544 0x0800 BS RAM = 128 0x7780 0x77FF	GS RAM = 28416 0x0800 BS RAM = 256 0x7700 0x77FF	GS RAM = 27648 0x0800 BS RAM = 1024 0x7400 0x77FF
RSS<1:0> = 10 AND RL_SSR = 0 OR RSS<1:0> = 01 AND RL_SSR = 1	GS RAM = 28416 0x0800 SS RAM = 256 0x7700 0x77FF	GS RAM = 28416 0x0800 SS RAM = 128 0x7700 BS RAM = 128 0x7780 0x77FF	GS RAM = 28416 0x0800 BS RAM = 256 0x7700 0x77FF	GS RAM = 27648 0x0800 BS RAM = 1024 0x7400 0x77FF
RSS<1:0> = 01 AND RL_SSR = 0 OR RSS<1:0> = 00 AND RL_SSR = 1	GS RAM = 26624 0x0800 SS RAM = 2048 0x7000 0x77FF	GS RAM = 26624 0x0800 SS RAM = 1920 0x7000 BS RAM = 128 0x7780 0x77FF	GS RAM = 26624 0x0800 SS RAM = 1792 0x7000 BS RAM = 256 0x7700 0x77FF	GS RAM = 26624 0x0800 SS RAM = 1024 0x7000 BS RAM = 1024 0x7400 0x77FF
RSS<1:0> = 00 AND RL_SSR = 0	GS RAM = 24576 0x0800 SS RAM = 4096 0x6800 0x77FF	GS RAM = 24576 0x0800 SS RAM = 3968 0x6800 BS RAM = 128 0x7780 0x77FF	GS RAM = 24576 0x0800 SS RAM = 3840 0x6800 BS RAM = 256 0x7700 0x77FF	GS RAM = 24576 0x0800 SS RAM = 3072 0x6800 BS RAM = 1024 0x7400 0x77FF

凡例 : OR = 論理和、AND = 論理積

Note: ブートセグメントにセキュア セグメント以上の容量を定義した場合、セキュア セグメントの容量選択は無効となり、セキュア セグメント自体も無効化されます。

表 23-4: 16 KB RAM を搭載したデバイスのデータ RAM のセグメント容量

コンフィグレーション ビット	RBS<1:0> = 11 OR RBS<1:0> = 10 AND RL_BSR = 1	RBS<1:0> = 10 AND RL_BSR = 0 OR RBS<1:0> = 01 AND RL_BSR = 1	RBS<1:0> = 01 AND RL_BSR = 0 OR RBS<1:0> = 00 AND RL_BSR = 1	RBS<1:0> = 00 AND RL_BSR = 0
RSS<1:0> = 11 OR RSS<1:0> = 10 AND RL_SSR = 1				
RSS<1:0> = 10 AND RL_SSR = 0 OR RSS<1:0> = 01 AND RL_SSR = 1				
RSS<1:0> = 01 AND RL_SSR = 0 OR RSS<1:0> = 00 AND RL_SSR = 1				
RSS<1:0> = 00 AND RL_SSR = 0				

凡例 : OR = 論理和、AND = 論理積

Note: ブートセグメントにセキュア セグメント以上の容量を定義した場合、セキュア セグメントの容量選択は無効となり、セキュア セグメント自体も無効化されます。

表 23-5: 8 KB RAM を搭載したデバイスのデータ RAM のセグメント容量

コンフィグレーション ビット	RBS<1:0> = 11 OR RBS<1:0> = 10 AND RL_BSR = 1	RBS<1:0> = 10 AND RL_BSR = 0 OR RBS<1:0> = 01 AND RL_BSR = 1	RBS<1:0> = 01 AND RL_BSR = 0 OR RBS<1:0> = 00 AND RL_BSR = 1	RBS<1:0> = 00 AND RL_BSR = 0
RSS<1:0> = 11 OR RSS<1:0> = 10 AND RL_SSR = 1	GS RAM = 6144 0x0800 0x1FFF	GS RAM = 6016 0x0800 BS RAM = 128 0x1F80 0x1FFF	GS RAM = 5888 0x0800 BS RAM = 256 0x1F00 0x1FFF	GS RAM = 5120 0x0800 BS RAM = 1024 0x1C00 0x1FFF
RSS<1:0> = 10 AND RL_SSR = 0 OR RSS<1:0> = 01 AND RL_SSR = 1	GS RAM = 5888 0x0800 SS RAM = 256 0x1F00 0x1FFF	GS RAM = 5888 0x0800 SS RAM = 128 0x1F00 BS RAM = 128 0x1F80 0x1FFF	GS RAM = 5888 0x0800 BS RAM = 256 0x1F00 0x1FFF	GS RAM = 5120 0x0800 BS RAM = 1024 0x1C00 0x1FFF
RSS<1:0> = 01 AND RL_SSR = 0 OR RSS<1:0> = 00 AND RL_SSR = 1	GS RAM = 4096 0x0800 SS RAM = 2048 0x1800 0x1FFF	GS RAM = 4096 0x0800 SS RAM = 1920 0x1800 BS RAM = 128 0x1F80 0x1FFF	GS RAM = 4096 0x0800 SS RAM = 1792 0x1800 BS RAM = 256 0x1F00 0x1FFF	GS RAM = 4096 0x0800 SS RAM = 1024 0x1800 BS RAM = 1024 0x1C00 0x1FFF
RSS<1:0> = 00 AND RL_SSR = 0	GS RAM = 2048 0x0800 SS RAM = 4096 0x1000 0x1FFF	GS RAM = 2048 0x0800 SS RAM = 3968 0x1000 BS RAM = 128 0x1F80 0x1FFF	GS RAM = 2048 0x0800 SS RAM = 3840 0x1000 BS RAM = 256 0x1F00 0x1FFF	GS RAM = 2048 0x0800 SS RAM = 3072 0x1000 BS RAM = 1024 0x1C00 0x1FFF

凡例 : OR = 論理和、AND = 論理積

Note: ブートセグメントにセキュア セグメント以上の容量を定義した場合、セキュア セグメントの容量選択は無効となり、セキュア セグメント自体も無効化されます。

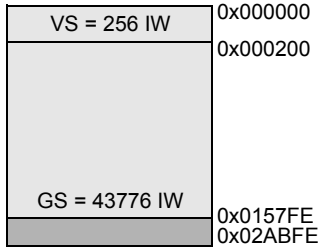
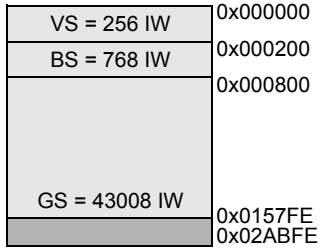
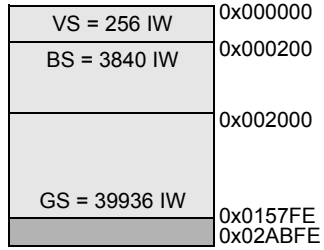
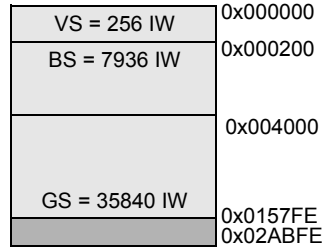
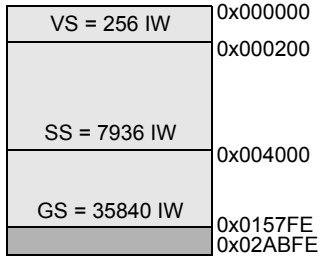
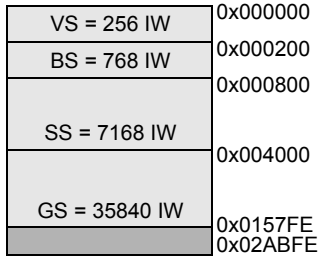
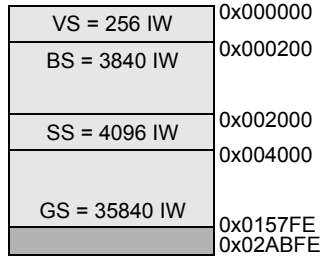
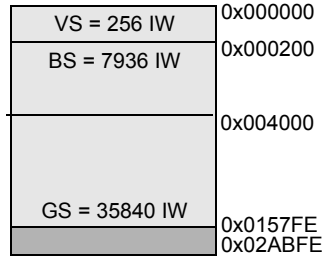
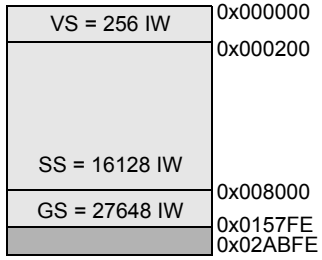
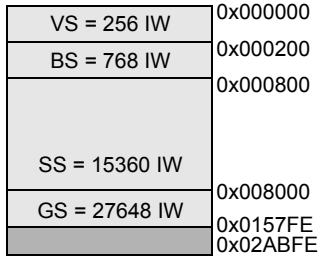
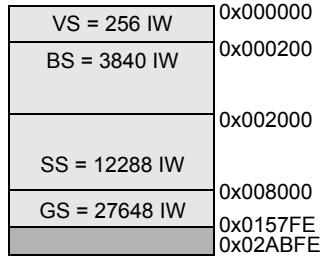
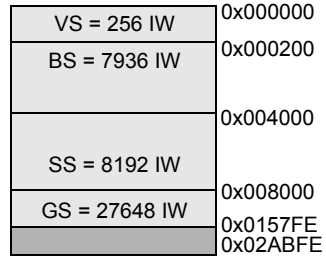
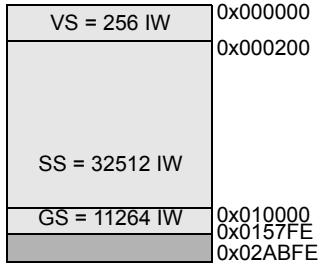
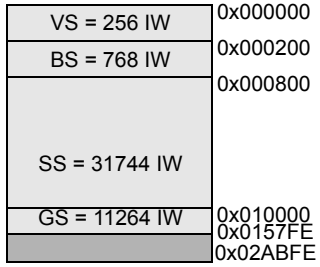
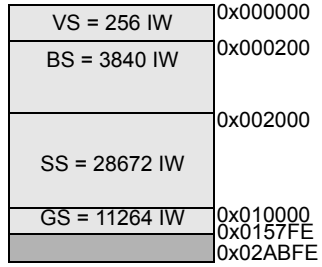
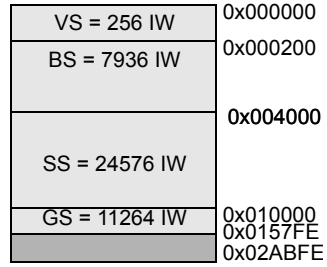
表 23-6: 256 KB デバイスのプログラム フラッシュのセグメント容量

コンフィグレーション ビット	BSS<2:0> = x11 0K	BSS<2:0> = x10 1K	BSS<2:0> = x01 4K	BSS<2:0> = x00 8K
SSS<2:0> = x11 0K	VS = 256 IW 0x000000 0x000200 GS = 87296 IW 0x02ABFE	VS = 256 IW 0x000000 0x000200 BS = 768 IW 0x000800 GS = 86528 IW 0x02ABFE	VS = 256 IW 0x000000 0x000200 BS = 3840 IW 0x002000 GS = 83456 IW 0x02ABFE	VS = 256 IW 0x000000 0x000200 BS = 7936 IW 0x004000 GS = 79360 IW 0x02ABFE
SSS<2:0> = x10 8K	VS = 256 IW 0x000000 0x000200 SS = 7936 IW 0x004000 GS = 79360 IW 0x02ABFE	VS = 256 IW 0x000000 0x000200 BS = 768 IW 0x000800 SS = 7168 IW 0x004000 GS = 79360 IW 0x02ABFE	VS = 256 IW 0x000000 0x000200 BS = 3840 IW 0x002000 SS = 4096 IW 0x004000 GS = 79360 IW 0x02ABFE	VS = 256 IW 0x000000 0x000200 BS = 7936 IW 0x004000 GS = 79360 IW 0x02ABFE
SSS<2:0> = x01 16K	VS = 256 IW 0x000000 0x000200 SS = 16128 IW 0x008000 GS = 71168 IW 0x02ABFE	VS = 256 IW 0x000000 0x000200 BS = 768 IW 0x000800 SS = 15360 IW 0x008000 GS = 71168 IW 0x02ABFE	VS = 256 IW 0x000000 0x000200 BS = 3840 IW 0x002000 SS = 12288 IW 0x008000 GS = 71168 IW 0x02ABFE	VS = 256 IW 0x000000 0x000200 BS = 7936 IW 0x004000 SS = 8192 IW 0x008000 GS = 71168 IW 0x02ABFE
SSS<2:0> = x00 32K	VS = 256 IW 0x000000 0x000200 SS = 32512 IW 0x010000 GS = 54784 IW 0x02ABFE	VS = 256 IW 0x000000 0x000200 BS = 768 IW 0x000800 SS = 31744 IW 0x010000 GS = 54784 IW 0x02ABFE	VS = 256 IW 0x000000 0x000200 BS = 3840 IW 0x002000 SS = 28672 IW 0x010000 GS = 54784 IW 0x02ABFE	VS = 256 IW 0x000000 0x000200 BS = 7936 IW 0x004000 SS = 24576 IW 0x010000 GS = 54784 IW 0x02ABFE

凡例 : IW = 命令ワード

Note: ブートセグメントにセキュア セグメント以上の容量を定義した場合、セキュア セグメントの容量選択は無効となり、セキュア セグメント自体も無効化されます。

表 23-7: 128 KB デバイスのプログラム フラッシュのセグメント容量

コンフィグレーション ビット	BSS<2:0> = x11 0K	BSS<2:0> = x10 1K	BSS<2:0> = x01 4K	BSS<2:0> = x00 8K
SSS<2:0> = x11 0K				
SSS<2:0> = x10 8K				
SSS<2:0> = x01 16K				
SSS<2:0> = x00 32K				

凡例 : IW = 命令ワード

Note: ブートセグメントにセキュア セグメント以上の容量を定義した場合、セキュア セグメントの容量選択は無効となり、セキュア セグメント自体も無効化されます。

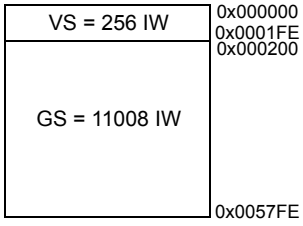
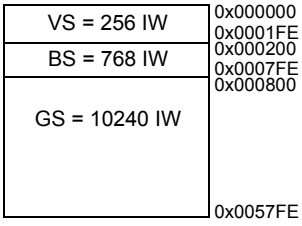
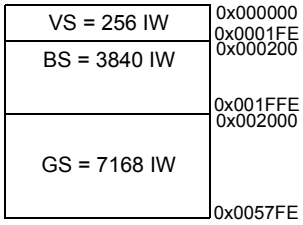
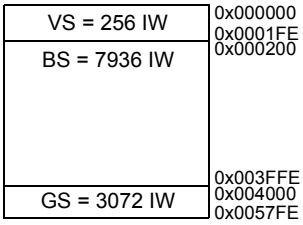
表 23-8: 64 KB デバイスのプログラム フラッシュのセグメント容量

コンフィグレーション ビット	BSS<2:0> = x11 0K	BSS<2:0> = x10 1K	BSS<2:0> = x01 4K	BSS<2:0> = x00 8K
SSS<2:0> = x11 0K	VS = 256 IW GS = 21760 IW 0x000000 0x00ABFE 0x02ABFE	VS = 256 IW BS = 768 IW GS = 20992 IW 0x000000 0x000200 0x000800 0x00ABFE 0x02ABFE	VS = 256 IW BS = 3840 IW GS = 17920 IW 0x000000 0x000200 0x002000 0x00ABFE 0x02ABFE	VS = 256 IW BS = 7936 IW GS = 13824 IW 0x000000 0x000200 0x004000 0x00ABFE 0x02ABFE
SSS<2:0> = x10 4K	VS = 256 IW SS = 3840 IW GS = 17920 IW 0x000000 0x000200 0x002000 0x00ABFE 0x02ABFE	VS = 256 IW BS = 768 IW SS = 3072 IW GS = 17920 IW 0x000000 0x000200 0x000800 0x002000 0x00ABFE 0x02ABFE	VS = 256 IW BS = 3840 IW GS = 17920 IW 0x000000 0x000200 0x002000 0x00ABFE 0x02ABFE	VS = 256 IW BS = 7936 IW GS = 13824 IW 0x000000 0x000200 0x004000 0x00ABFE 0x02ABFE
SSS<2:0> = x01 8K	VS = 256 IW SS = 7936 IW GS = 13824 IW 0x000000 0x000200 0x004000 0x00ABFE 0x02ABFE	VS = 256 IW BS = 768 IW SS = 7168 IW GS = 13824 IW 0x000000 0x000200 0x000800 0x004000 0x00ABFE 0x02ABFE	VS = 256 IW BS = 3840 IW SS = 4096 IW GS = 13824 IW 0x000000 0x000200 0x002000 0x004000 0x00ABFE 0x02ABFE	VS = 256 IW BS = 7936 IW GS = 13824 IW 0x000000 0x000200 0x004000 0x00ABFE 0x02ABFE
SSS<2:0> = x00 16K	VS = 256 IW SS = 16128 IW GS = 5632 IW 0x000000 0x000200 0x008000 0x00ABFE 0x02ABFE	VS = 256 IW BS = 768 IW SS = 15360 IW GS = 5632 IW 0x000000 0x000200 0x000800 0x008000 0x00ABFE 0x02ABFE	VS = 256 IW BS = 3840 IW SS = 12288 IW GS = 5632 IW 0x000000 0x000200 0x002000 0x008000 0x00ABFE 0x02ABFE	VS = 256 IW BS = 7936 IW SS = 8192 IW GS = 5632 IW 0x000000 0x000200 0x004000 0x008000 0x00ABFE 0x02ABFE

凡例 : IW = 命令ワード

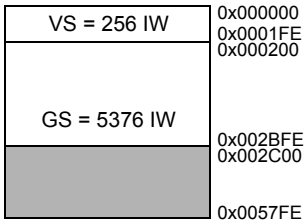
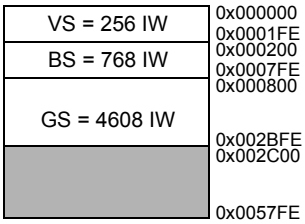
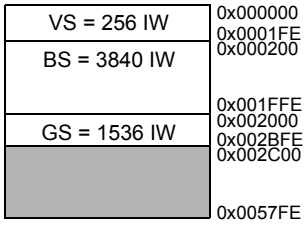
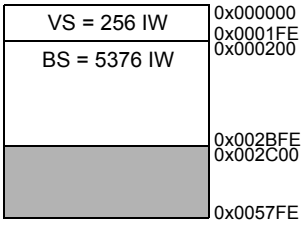
Note: ブートセグメントにセキュア セグメント以上の容量を定義した場合、セキュア セグメントの容量選択は無効となり、セキュア セグメント自体も無効化されます。

表 23-9: 32 KB デバイスのプログラム フラッシュのセグメント容量

BSS<2:0> = x11 0K	BSS<2:0> = x10 1K
	
BSS<2:0> = x01 4K	**BSS<2:0> = x00 8K**
	

凡例 : IW = 命令ワード

表 23-10: 16 KB デバイスのプログラム フラッシュのセグメント容量

BSS<2:0> = x11 0K	BSS<2:0> = x10 1K
	
BSS<2:0> = x01 4K	**BSS<2:0> = x00 8K**
	

凡例 : IW = 命令ワード

表 23-11: 12 KB デバイスのプログラム フラッシュのセグメント容量

コンフィグレーション ビット	セグメント容量
BSS<2:0> = x11 0K	VS = 256 IW GS = 3840 IW 0x000000 0x0001FE 0x000200 0x001FFE
BSS<2:0> = x10 256	VS = 256 IW BS = 256 IW GS = 3584 IW 0x000000 0x0001FE 0x000200 0x0003FE 0x000400 0x001FFE
BSS<2:0> = x01 768	VS = 256 IW BS = 768 IW GS = 3072 IW 0x000000 0x0001FE 0x000200 0x0007FE 0x000800 0x001FFE
BSS<2:0> = x00 1792	VS = 256 IW BS = 1792 IW GS = 2048 IW 0x000000 0x0001FE 0x000200 0x000FFE 0x001000 0x001FFE

凡例 : IW = 命令ワード

Note: 表 23-3 から表 23-11 に示したセグメント構成は代表例であり、これらとは異なる構成を持つデバイスもあります。各種設定に対するメモリセグメントの容量は、各デバイスのデータシートで確認してください。

23.6.2 ブートセグメントのセキュリティ レベル選択

ブートセグメントのセキュリティ レベルはコンフィグレーション ビット BSS2 (FBS<3>) によって、以下のように設定されます。

- 1 = 標準セキュリティ
- 0 = 高セキュリティ

ブートセグメントを高セキュリティに設定すると、標準セキュリティに比べてアクセス方法の数が制限されます。両設定の差異は後述します。詳細は **23.11「プログラムフローに関するルール」**を参照してください。

23.6.3 ブートセグメントの書き込み保護

コンフィグレーション ビットBWRP (FBS<0>)をプログラムする事でブートセグメントに書き込み保護を設定できます。

- 1 = ブートセグメントへの書き込み許可
- 0 = ブートセグメントへの書き込み禁止

書き込み保護を設定すると、プログラム フラッシュのブートセグメントに対するページ消去 / 書き込み動作が無効になります。特殊機能レジスタ NVMCON の WR ビットをセットしても、書き込み動作は開始しません。ただし、ブートセグメント全体に対する消去動作は禁止されず、これを実行した場合、セキュア セグメントと汎用セグメントも消去されます。

23.6.4 ブートセグメント RAM の割り当て

デバイスのデータ RAM メモリの一部もブートセグメントに割り当てる事ができます。この場合、ブートセグメント内で実行されるコードだけが、この領域にアクセスできます。これにより、ブートセグメント内で実行されるアルゴリズムのデータの整合性を確保できます。

ブートセグメントを割り当てない場合 (BSS<2:0> = x11 (FBS<3:1>))、RAM セグメントを割り当てる事ができません。ブートセグメント RAM の有無と容量は、コンフィグレーション ビット RBS<1:0> (FBS<7:6>) によって指定します。

ブートセグメント RAM を割り当てない設定もあります。これは消去され未プログラム状態のデバイスの既定値です。

図 23-4 に示す通り、ブートセグメント RAM はデータ RAM の末尾または DMA メモリの直前を占有します。ブートセグメント RAM は RBS<1:0> ビットによって指定されるアドレスから開始します。

Note:	DMA RAM は全ての dsPIC33F に実装されているわけではありません。DMA RAM の有無と容量については各デバイスのデータシートを参照してください。

図 23-4: ブートセグメント データ RAM の割り当て

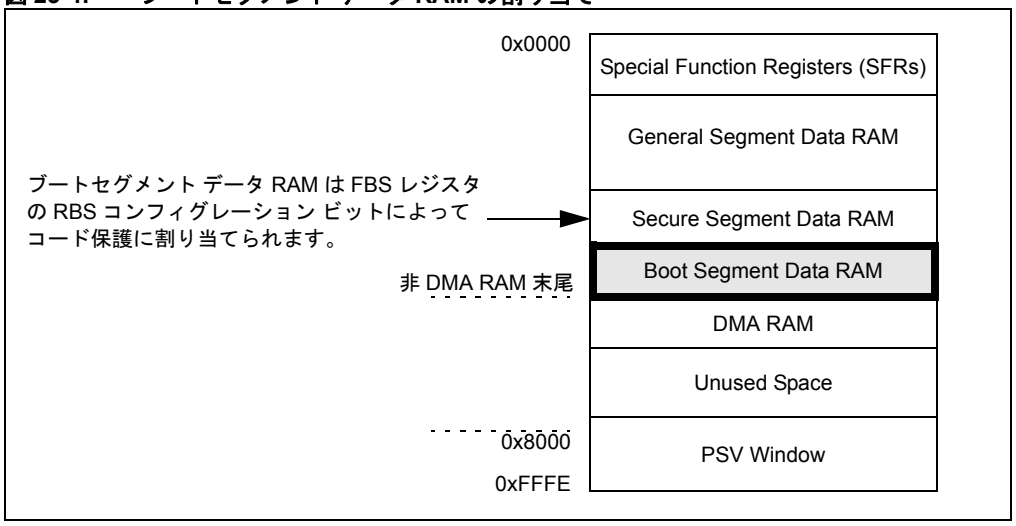


表 23-12 に dsPIC33F のブートセグメント RAM 容量の例を示します。ここに示す開始アドレスは代表例です。実際のアドレスは、各デバイスのデータシートを参照してください。

表 23-12: ブートセグメント RAM 容量の例

RBS<1:0>	BS 容量 / バイト	BS 開始アドレス	BS 終了アドレス
11	ブートセグメントなし		
10	小 /128	EOM-0x007F	EOM
01	中 /256	EOM-0x00FF	EOM
00	大 /1024	EOM-0x03FF	EOM

Note: EOM は DMA RAM を除くデータ RAM の末尾位置を意味します。

23.6.5 ブートセグメント RAM の実行時解放

ブートセグメント内のアルゴリズムの実行が完了し、より優先度が低いセグメント内のコードに実行を戻す場合、ブートセグメントに割り当てられた RAM の一部を解放すると有利な場合があります。そのために用意されているビットがブートセグメント RAM を制御する特殊機能レジスタに含まれる RL_BSR (BSRAM<0>) ビットです (レジスタ 23-2 参照)。このビットをセットすると、システムは BS RAM の一部を BS の次に優先度の高い定義済みセグメントに解放します。表 23-13 は、RL_BSR =「0」と RL_BSR =「1」の場合の RAM のマッピング例です。

表 23-13: ブートセグメント RAM の解放

RBS<1:0>	下記ビット設定時のブートセグメント容量	
	RL_BSR = 0	RL_BSR = 1
11	ブートセグメントなし	
10	小	ブートセグメントなし
01	中	小
00	大	中

23.6.6 セキュア RAM の一般用途への解放

セキュアコード セグメントは、割り当てられたセキュア RAM の一部を、動作中いつでも一般用途に解放できます。例えば、汎用セグメントのコード実行中は、機密性の高い揮発性の変数を格納するために最低限の容量をブートセグメント用に確保しておきます。続いて、コードがブートセグメントに分岐してアルゴリズムを実行する時点で、ブートセグメントのコードによって RL_BSR ビットをクリアして、セキュア RAM を最大割り当て量まで確保します。ブートセグメントのコード実行が完了したら、再び RL_BSR ビットをセットしてセキュア RAM の割り当て量を必要最低限に戻す事もできます。

ブートセグメントとセキュア セグメントには、それぞれ RL_BSR ビットと RL_SSR ビットを含む BSRAM レジスタと SSRAM レジスタが関連付けられています。BSRAM レジスタへの書き込みアクセス権は BS のみ、SSRAM レジスタへの書き込みアクセス権は SS のみが持ちます。

Note: 全てのリセットは、RL_BSR ビットと RL_SSR ビットをクリアするため、リセット後は最大割り当て領域が保護された状態になります。

RAM セキュリティ ビットは RAM が保護状態であるかどうかを決定します。

- RSS<1:0> = 11 (FSS<7:6>) の場合、セキュア RAM は割り当てられず、RL_SSR ビットは無視されます。
- RBS<1:0> = 11 (FBS<7:6>) の場合、ブート RAM は割り当てられず、RL_BSR ビットは無視されます。

レジスタ 23-2: BSRAM: ブートセグメント RAM 特殊機能レジスタ

U-0	U-0	U-0	U-0	U-0	U-0	U-0	U-0
—	—	—	—	—	—	—	—
bit 15							bit 8

U-0	U-0	U-0	U-0	U-0	R-0	R-0	R/W-0
—	—	—	—	—	IW_BSR	IR_BSR	RL_BSR
bit 7							bit 0

凡例 :			
R = 読み出し可能ビット	W = 書き込み可能ビット	U = 未実装ビット、「0」として読み出し	
-n = POR 時の値	「1」= ビットをセット	「0」= ビットをクリア	x = ビットは未知

bit 15-3 **未実装 :** 「0」として読み出し

bit 2 **IW_BSR:** ブートセグメント RAM 不正書き込みステータスビット (読み出し専用)
1 = このレジスタの前回読み出し以降、1 回以上不正な書き込みが試行された
0 = このレジスタの前回読み出し以降、不正な書き込みは 1 回も施行されていない

IW_BSR ビットは、全てのリセットでクリアされます。また、BS 内のコード実行中に BSRAM レジスタを読み出した*後*にもクリアされます。

bit 1 **IR_BSR:** ブートセグメント RAM 無効読み出しステータスビット (読み出し専用)
1 = このレジスタの前回読み出し以降、1 回以上無効な読み出しが発生した
0 = このレジスタの前回読み出し以降、保護された BS RAM 部分に対する無効読み出しは一切発生していない

IR_BSR ビットは、全てのリセットでクリアされます。また、BS 内のコード実行中に BSRAM レジスタを読み出した後にもクリアされます。

bit 0 **RL_BSR:** ブートセグメント RAM 解放ステータスビット
1 = BSがセキュアRAMを一般用途向けに解放した (最上位128バイト以外の全領域が解放されます)
0 = BSRAM は BS のみが使用できるように保護されている

RL_BSR ビットは、全てのリセットでゼロにクリアされます。

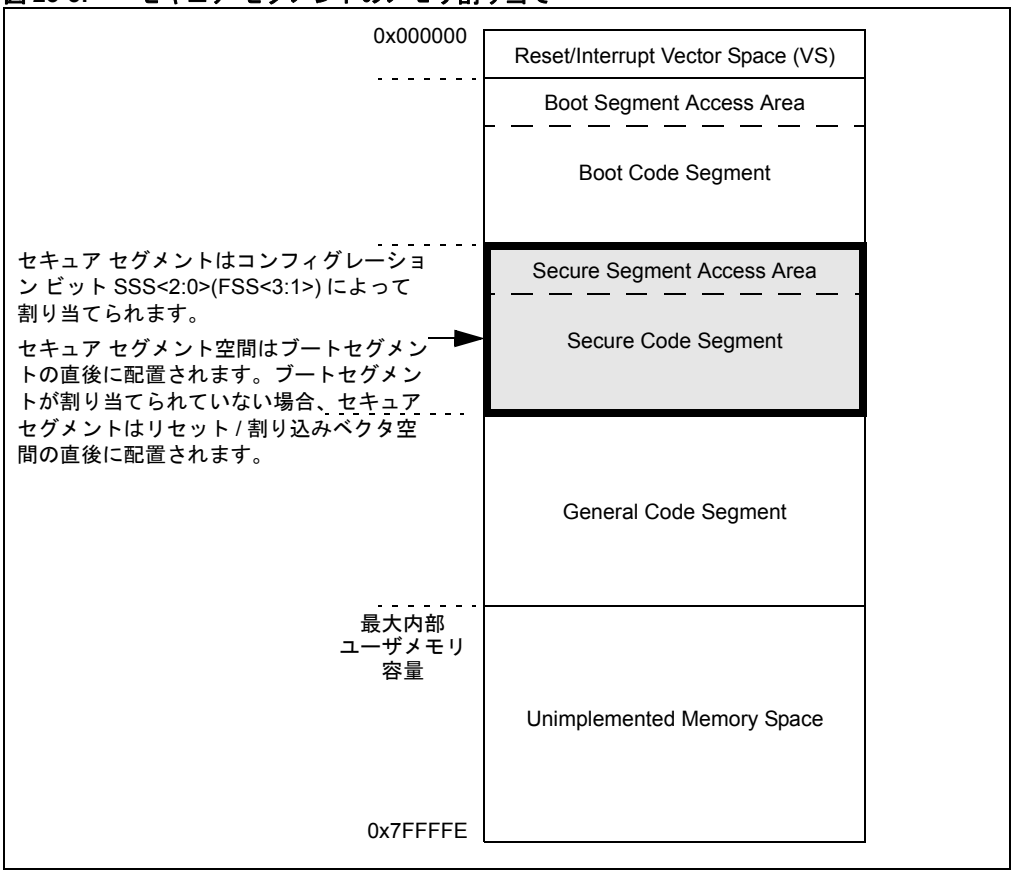
23.7 セキュア セグメント (SS)

セキュア セグメントは、ブートセグメントに次いで 2 番目に高い権限を持ち、非公開アルゴリズム ルーチンを格納するのに最適です。より優先度の低いセグメントからセキュア セグメントへのアクセスは「呼び出し」に限られます。

23.7.1 セキュア セグメントの割り当て

セキュア セグメントはコンフィグレーション ビット SSS<2:0> (FSS<3:1>) によって割り当てられます。図 23-5 に示す通り、セキュア セグメントはブートセグメントの直後から開始します。ブートセグメントが定義されていない場合、リセット / 割り込みベクタ空間の終端から開始します。既定値ではセキュア セグメントは割り当てられません。

図 23-5: セキュア セグメントのメモリ割り当て



セキュア セグメントは SSS<2:0> ビットによって指定されるアドレスまで続きます。表 23-14 にセキュア セグメント容量の例を示します (64 KB フラッシュメモリ搭載デバイスの場合)。ここに示した終了アドレスは代表例です。デバイス固有のアドレスは、各デバイスのデータシートを参照してください。

表 23-14: セキュア セグメント容量の例

SSS<2:0>	セキュリティ レベル	SS 容量	SS 開始アドレス	SS 終了アドレス
x11	セキュア プログラム フラッシュ セグメントなし			
110	標準	小	E.O. BS + 1	0x001FFE
010	高	小	E.O. BS + 1	0x001FFE
101	標準	中	E.O. BS + 1	0x003FFE
001	高	中	E.O. BS + 1	0x003FFE
100	標準	大	E.O. BS + 1	0x007FFE
000	高	大	E.O. BS + 1	0x007FFE

Note: E.O. BS はブートセグメントの終端位置を表します。

レジスタ 23-3: FSS: セキュア セグメント コンフィグレーション レジスタ バイト

R/P	R/P	r	r	R/P	R/P	R/P	R/P
RSS<1:0>(1)		—	—	SSS<2:0>(2)			SWRP(3)
bit 7							bit 0

凡例 :	r = 予約	P = プログラム可能ビット
R = 読み出し可能ビット	W = 書き込み可能ビット	U = 未実装ビット、「0」として読み出し
-n = POR 時の値	「1」= ビットをセット	「0」= ビットをクリア x = ビットは未知

bit 7-6 **RSS<1:0>:** セキュア セグメント RAM コード保護ビット (1)

11 = セキュア RAM を定義しない
10 = BS RAM より 256 バイト少ないセキュア RAM を定義する
01 = BS RAM より 2048 バイト少ないセキュア RAM を定義する
00 = BS RAM より 4096 バイト少ないセキュア RAM を定義する

bit 5-4 **予約 :** プログラミング値は無効になります。

bit 3-1 **SSS<2:0>:** セキュア セグメント プログラム フラッシュ コード保護ビット (2)

x11 = セキュア プログラム フラッシュ セグメントを定義しない
110 = 標準セキュリティ、小容量セキュア セグメント
010 = 高セキュリティ、小容量セキュア セグメント
101 = 標準セキュリティ、中容量セキュア セグメント
001 = 高セキュリティ、中容量セキュア セグメント
100 = 標準セキュリティ、大容量セキュア セグメント
000 = 高セキュリティ、大容量セキュア セグメント

デバイス固有の情報は表 23-8 を参照してください。

bit 0 **SWRP:** セキュア セグメント プログラム フラッシュ書き込み保護ビット (3)

1 = セキュア セグメントへの書き込み許可
0 = セキュア セグメントへの書き込み禁止

Note 1: セキュア セグメント RAM コード保護機能は全てのデバイスに搭載されているわけではありません。デバイス固有の情報は、表 23-3、表 23-4、表 23-5 を参照してください。

2: セキュア セグメント容量の大、中、小の厳密な定義はデバイスごとに異なります。デバイス固有の情報は、表 23-6、表 23-7、表 23-8 を参照してください。

3: セキュア セグメントが不要の場合、SWRP ビットを「1」にプログラムする必要があります。

23.7.2 セキュア セグメントのセキュリティ レベル選択

セキュアコード セグメントのセキュリティ レベルはコンフィグレーション ビット SSS2 (FSS<3>) によって、以下のように設定されます。

- 1 = 標準セキュリティ
- 0 = 高セキュリティ

セキュア セグメントを高セキュリティに設定すると、標準セキュリティに比べてアクセス方法の数が制限されます。両設定の差異は後述します。

23.7.3 セキュア セグメントの書き込み保護

コンフィグレーション ビット SWRP (FSS<0>) をプログラムする事でセキュア セグメントに書き込み保護を設定できます。

- 1 = セキュア セグメントへの書き込み許可
- 0 = セキュア セグメントへの書き込み禁止

書き込み保護を設定すると、プログラム フラッシュのセキュア セグメントに対するページ消去動作とプログラミング動作が無効化されます。特殊機能レジスタ NVMCON の WR ビットをセットしてもこの動作は開始されません。ただし、セキュア セグメント全体に対する消去動作は禁止されず、これを実行した場合、汎用セグメントも消去されます。

23.7.4 セキュア セグメント RAM の割り当て

ブートセグメントと同様に、セキュア セグメントにもデータ RAM の一部を割り当ててコードを保護できます。ただし、セキュア セグメントを割り当てていない場合、つまり SSS<2:0> = x11 (FSS<3:1>) の場合、RAM セグメントを割り当てる事ができません。セキュア セグメント RAM の有無と容量は、コンフィグレーション ビット RSS<1:0> (FSS<7:6>) によって指定します。

セキュア セグメント RAM を使用しない設定もあります。これは、消去され未プログラム状態のデバイスの既定値です。

セキュア セグメント RAM は、ブートセグメント RAM の直前で終了します。一方、その開始位置は RSS<1:0> ビットによって指定されるアドレスです。

図 23-6: セキュア セグメント データ RAM の割り当て

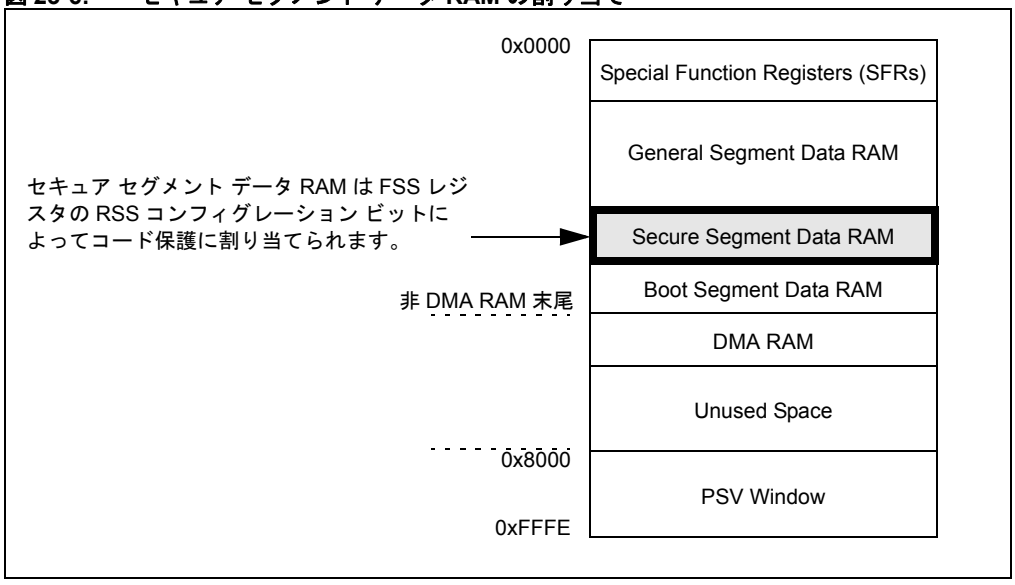


表 23-15 にセキュア セグメント RAM の割り当ての例を示します。ここに示した開始アドレスは代表例です。デバイス固有のアドレスは、各デバイスのデータシートを参照してください。

表 23-15: セキュア セグメント RAM 容量の例

RSS<1:0>	SS 容量	SS 開始アドレス	SS 終了アドレス
11	セキュア セグメントなし		
10	小	S.O. BS - 0x0100	S.O. BS - 1
01	中	S.O. BS - 0x0800	S.O. BS - 1
00	大	S.O. BS - 0x1000	S.O. BS - 1

Note: S.O. BS はブートセグメント データ RAM の開始位置を表します。

23.7.5 セキュア セグメント RAM の実行時解放

ブートセグメントと同様、セキュア セグメントも RAM の割り当てと解放を制御できます。その目的でセキュア セグメント RAM を制御する特殊機能レジスタ SSRAM に RL_SSR (SSRAM<0>) ビットが用意されています (レジスタ 23-4 参照)。このビットをセットすると、システムはセキュア セグメント (SS) RAM の一部を、次に優先度の高い定義済みセグメントに解放します。表 23-16 は、RL_SSR = 0 と RL_SSR = 1 の場合の RAM のマッピング例です。

表 23-16: セキュア セグメント RAM の解放

RSS<1:0>	下記ビット設定時のセキュア セグメント容量	
	RL_SSR = 0	RL_SSR = 1
11	セキュア セグメントなし	
10	小	セキュア セグメントなし
01	中	小
00	大	中

23.7.6 セキュア セグメント RAM の一般用途への解放

セキュアコード セグメントは、割り当てられたセキュア RAM の一部を、動作中いつでも一般用途に解放できます。例えば、汎用セグメントのコード実行中は、機密性の高い揮発性の変数を格納するために最低限の容量をセキュア セグメント用に確保しておきます。続いて、コードがセキュア セグメントに分岐してアルゴリズムを実行する時点で、セキュア セグメントのコードによって RL_SSR ビットをクリアして、セキュア RAM を最大割り当て量まで確保します。セキュア セグメントのコード実行が完了したら、再び RL_SSR ビットをセットしてセキュア RAM の割り当て量を必要最低限に戻す事もできます。

ブートセグメントとセキュア セグメントには、それぞれ RL_BSR ビットと RL_SSR ビットを含む BSRAM レジスタと SSRAM レジスタが関連付けられています。BSRAM レジスタへの書き込みアクセス権はブートセグメントのみ、SSRAM レジスタへの書き込みアクセス権はセキュア セグメントのみが持ちます。

全てのリセットは、RL_SSR ビットをクリアするため、リセット後は最大割り当て領域が保護された状態になる事に注意してください。

RAM セキュリティ ビットは RAM が保護状態であるかどうかを決定します。RSS<1:0> = 11 (FSS<7:6>) の場合、セキュア RAM は割り当てられず、RL_SSR ビットは無視されます。

レジスタ 23-4: SSRAM: セキュア セグメント RAM 特殊機能レジスタ

U-0	U-0	U-0	U-0	U-0	U-0	U-0	U-0
—	—	—	—	—	—	—	—
bit 15							bit 8

U-0	U-0	U-0	U-0	U-0	U-0	R-0	R-0
—	—	—	—	—	IW_SSR	IR_SSR	RL_SSR
bit 7							bit 0

凡例 :			
R = 読み出し可能ビット	W = 書き込み可能ビット	U = 未実装ビット、「0」として読み出し	
-n = POR 時の値	「1」= ビットをセット	「0」= ビットをクリア	x = ビットは未知

bit 15-3 **未実装 :** 「0」として読み出し

bit 2 **IW_SSR:** セキュア セグメント RAM 不正書き込みステータスビット (読み出し専用)
1 = このレジスタの前回読み出し以降、1 回以上不正な書き込みが試行された
0 = このレジスタの前回読み出し以降、保護された SSRAM に対する不正な書き込みは 1 回も試行されていない

IW_SSR ビットは、全てのリセットでクリアされます。また、セキュア セグメント内のコード実行中に SSRAM レジスタを読み出した*後*にもクリアされます。

bit 1 **IR_SSR:** セキュア セグメント RAM 無効読み出しステータスビット (読み出し専用)
1 = このレジスタの前回読み出し以降、1 回以上無効な読み出しが発生した
0 = このレジスタの前回読み出し以降、保護された SSRAM に対する無効読み出しは一切発生していない

IR_SSR ビットは、全てのリセットでクリアされます。また、セキュア セグメント内のコード実行中に SSRAM レジスタを読み出した*後*にもクリアされます。

bit 0 **RL_SSR:** セキュア セグメント解放ステータスビット
1 = セキュア セグメントがセキュア RAM を一般用途向けに解放した (最上位 128 バイト以外の全領域が解放されます)
0 = SSRAM はセキュア セグメントのみが使用できるように保護されている

RL_SSR ビットは、全てのリセットでゼロにクリアされます。

23.8 汎用セグメント (GS)

汎用セグメントは最も低いセキュリティ権限を持ちます。アプリケーション コードの大部分を格納する事を目的としています。その容量は実質的に内蔵メモリ容量からブートセグメントとセキュア セグメントの容量を差し引いたものです。ブートセグメントまたはセキュア セグメントが存在しない場合、汎用セグメントが内蔵メモリの全てを使用します。

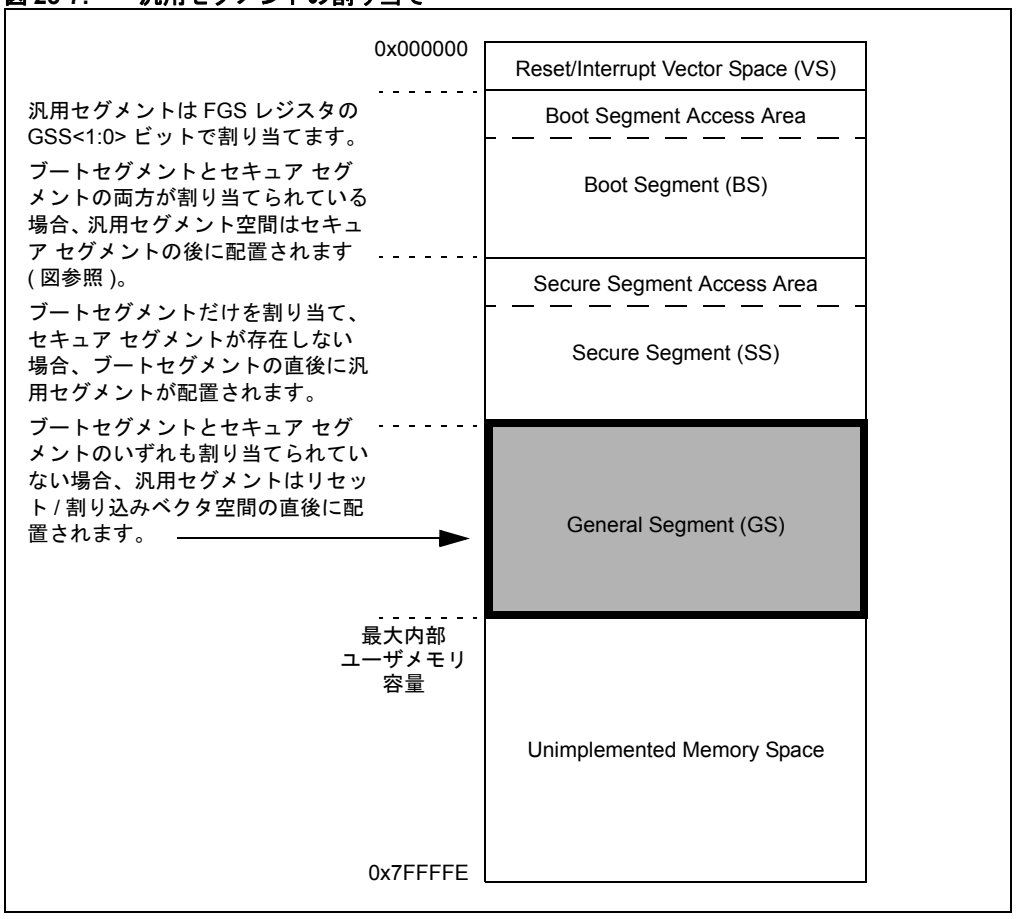
23.8.1 汎用セグメントの割り当て

汎用セグメントは、ブートセグメントとセキュア セグメントを割り当てたかどうかにかかわらず、常に存在します。設定には FGS レジスタの GSS<1:0> コンフィグレーション ビットを使用します。汎用セグメントの位置は、ブートセグメントとセキュア セグメントの有無によって変化します。

図 23-7 に示すように、ブートセグメントとセキュア セグメントの両方が割り当てられている場合、汎用セグメントはセキュア セグメントの直後に位置します。ブートセグメントだけを割り当て、セキュア セグメントが存在しない場合、ブートセグメントの直後に汎用セグメントが配置されます。ブートセグメントとセキュア セグメントのいずれも割り当てられていない場合、汎用セグメントはリセット / 割り込みベクタ空間 (VS) の直後に配置されます。

既定値ではブートセグメントとセキュア セグメントは割り当てられません。従って、既定値ではプログラムメモリ全体が汎用セグメントとして割り当てられます。

図 23-7: 汎用セグメントの割り当て



レジスタ 23-5: FGS: 汎用セグメント コンフィグレーション レジスタ

r	r	r	r	r	R/P	R/P	R/P
—	—	—	—	—	GSS<1:0>(1)		GWRP
bit 7							bit 0

凡例 :	r = 予約	P = プログラム可能ビット	
R = 読み出し可能ビット	W = 書き込み可能ビット	U = 未実装ビット、「0」として読み出し	
-n = POR 時の値	「1」= ビットをセット	「0」= ビットをクリア	x = ビットは未知

bit 7-3 **予約 :** 使用不可

bit 2-1 **GSS<1:0>:** 汎用セグメント プログラム フラッシュ コード保護ビット (1)

11 = 汎用セグメントを保護しない
10 = 標準セキュリティ (汎用プログラム フラッシュ セグメントは SS の終端から開始し、EOM で終了する)
0x = 高セキュリティ (汎用プログラム フラッシュ セグメントは SS の終端から開始し、EOM で終了する)

bit 0 **GWRP:** 汎用セグメント プログラム フラッシュ書き込み保護ビット

1 = 汎用セグメントへの書き込み許可
0 = 汎用セグメントへの書き込み禁止

Note 1: 一部のデバイスでは、GSS<1> が予約済みで、GSS<0> を GCP ビットとして使用します。

23.8.2 汎用セグメントのセキュリティ レベル選択

デバイスによっては、汎用セグメントに対して最大 3 段階のセキュリティ レベルを選択できます。使用できるオプションの数は、各デバイスのデータ シートを参照してください。

このセグメントの保護レベルは、コンフィグレーション ビット GSS<1:0> (FGS<2:1>) によって指定します。

- 11 = 保護なし
- 10 = 標準セキュリティ
- 0X = 高セキュリティ

23.8.3 汎用セグメントの書き込み保護

コンフィグレーション ビット GWRP (FGS<0>) を使うと、汎用セグメントにもブートセグメントと同様の書き込み保護を設定できます。

- 1 = 汎用コードセグメントへの書き込み許可
- 0 = 汎用コードセグメントへの書き込み禁止

23.9 リセット、トラップ、割り込みサービスルーチン (ISR) のベクタ空間

命令の先頭 256 ワードの領域は、RESET 命令、トラップ、割り込みのベクタ空間として予約済みです。

このセグメントに対する保護は、BSS<2:0> (FBS<3:1>) と GSS<1:0> (FGS<2:1>) または GCP (FGS<1>) コード保護ビットの状態によって決まります。ブートセグメントを割り当てた場合、ベクタ空間はブートセグメントと同じ設定で保護されます。言い換えれば、ブートセグメントを定義した場合、ベクタ空間の消去動作とプログラミング動作は、ブートセグメントのコードによってのみ実行できるという事です。ブートセグメントを割り当てない場合のベクタ空間の保護は汎用セグメントと同じになり、汎用セグメントのコードによってベクタ空間を消去およびプログラミングできます。

このセグメントへの書き込みは、ブートセグメントが割り当てられている場合は BWRP ビット、割り当てられていない場合は GWRP ビットで許可 / 禁止できます。

23.10 セキュリティ権限の定義

3 種類のコード保護セグメント間の相対的な権限レベルについて理解する事は重要です。動作の中には、対象となるセグメントとの間の相対的な権限の高低によって制限が課されるものがあります。例えば、ブートセグメントは最高レベルの権限を持ち、より権限の低いセグメント内のコードに直接アクセスできます。セキュア セグメントは汎用セグメント内のコードに直接アクセスできますが、ブートセグメント内のコードに対しては呼び出ししか実行できません。汎用セグメントがより高い権限を持つセグメントに含まれるコードにアクセスする手段は呼び出しのみです。

アクセス権の詳細は、**23.11「プログラムフローに関するルール」**から **23.14.1「RTSP モードによるデバイス プログラミングのルール」**で説明します。表 23-17 に、通常動作時におけるルールの概要を示します。

表 23-17: 権限が必要な動作の概要

ターゲット セグメント		汎用セグメント						セキュア セグメント				ブートセグメント				IVT と AIVT					
保護レベル		なし		標準		高		標準		高		標準		高		なし		標準		高	
書き込み保護		なし	あり	なし	あり	なし	あり	なし	あり	なし	あり	なし	あり	なし	あり	なし	あり	なし	あり	なし	あり
要求動作 (可 / 不可)																					
ターゲット セグメントへの PC ロールオーバー		可	可	可	可	可	可	可	可	可	可	N/A	N/A	N/A	N/A	N/A **(Note 3)**					
リセットベクタ命令によるターゲット セグメントへの PFC **(Note 5)**		可	可	可	可	可	可	可	可	**Note 2**	**Note 2**	可	可	**Note 2**	**Note 2**	**Note 4**					
ターゲット セグメントへの VFC (ベクタフロー変更) **(Note 5)**		可	可	可	可	可	可	可	可	**Note 2**	**Note 2**	可	可	**Note 2**	**Note 2**	**Note 4**					
BS からターゲット セグメントへの PFC **(Note 1)**		可	可	可	可	可	可	可	可	可	可	可	可	可	可	**Note 4**					
SS からターゲット セグメントへの PFC **(Note 1)**		可	可	可	可	可	可	可	可	可	可	可	可	**Note 2**	**Note 2**	**Note 4**					
GS からターゲット セグメントへの PFC **(Note 1)**		可	可	可	可	可	可	可	可	**Note 2**	**Note 2**	可	可	**Note 2**	**Note 2**	**Note 4**					
ターゲット セグメント RAM の R/W (右記のセグメントからの実行時) Note: アクセス対象のスタックが GS RAM 空間内にある事が前提	BS	可	可	可	可	可	可	不可	不可	不可	不可	可	可	可	可	−					
	SS	可	可	可	可	可	可	可	可	可	可	不可	不可	不可	不可						
	GS	可	可	可	可	可	可	不可	不可	不可	不可	不可	不可	不可	不可						
ターゲット セグメント プログラム フラッシュの テーブル読み出し / PSV (右記のセグメントからの実行時) **(Note 7)**	BS	可	可	可	可	不可	不可	可	可	不可	不可	可	可	可	可	可	可	可	可	可	可
	SS	可	可	可	可	不可	不可	可	可	可	可	不可	不可	不可	不可	可	可	可	可	可	可
	GS	可	可	可	可	可	可	不可	不可	不可	不可	不可	不可	不可	不可	可	可	可	可	可	可
ターゲット セグメントのテーブル書き込み (書き込みラッチへの読み込み)		可	可	可	可	可	可	可	可	可	可	可	可	可	可	可	可	可	可	可	可
ターゲット セグメント プログラム フラッシュの行プログラム / 消去 (右記のセグメントからの実行時)	BS	可	不可	可	不可	不可	不可	可	不可	不可	不可	可	不可	可	不可	可	不可	可	不可	**Note 8**	不可
	SS	可	不可	可	不可	不可	不可	可	不可	可	不可	不可	不可	不可	不可	**Note 6**	不可	**Note 6**	不可	**Note 6**	不可
	GS	可	不可	可	不可	可	不可	不可	不可	不可	不可	不可	不可	不可	不可	**Note 6**	不可	**Note 6**	不可	**Note 6**	不可
ターゲット セグメント データ フラッシュの消去 (右記のセグメントからの実行時)	BS	可	可	可	可	可	可	可	可	可	可	可	可	可	可	−					
	SS	可	可	可	可	可	可	可	可	可	可	可	可	可	可						
	GS	可	可	可	可	可	可	可	可	可	可	可	可	可	可						
全消去		RTSP モードでは全消去コマンドは無効																			

Note 1: プログラムフロー変更 (PFC) は、PC に通常の自動インクリメントされた値ではなく、他の新しい値が読み込まれる事と定義します。JUMP、CALL、RETURN、RETFIE、計算型ジャンプ等の命令があります。

2: PFC のターゲットとして許されるのはセグメント内の最初の 32 命令位置のみです。

3: VS セグメント内では実行が許可されていないため、このような条件はあり得ません。

4: IVT および AIVT セグメントへの PFC 動作 (すなわち分岐、呼び出し等) は可能です。ただし、左記のセグメントからこれを試みると、即座に不正アドレストラップが発生します (フロー変更先にアドレス 0x000000 のリセットベクタを指定した場合を除く)。

5: ベクタフロー変更 (VFC) は、PC に割り込みまたはトラップのベクタアドレスが読み込まれた場合と定義されます。

6: この動作は、より高いセキュリティ権限のセグメントが定義されていない場合にのみ許可されます。

7: TBLRD または DS 読み出しは実行可能です。ただし、許可されていない場合はオール「0」が返されます。

8: この動作が実行可能なデバイスは、dsPIC33FJXXXGPX06/X08/X10、dsPIC33FJXXXMCX06/X08/X10、PIC24HJXXXGPX06/X08/X10 のみです。その他の全てのデバイスでは、この動作は禁止されています。

表 23-17: 権限が必要な動作の概要 (続き)

ターゲット セグメント	汎用セグメント						セキュア セグメント				ブートセグメント				IVT と AIVT					
保護レベル	なし		標準		高		標準		高		標準		高		なし		標準		高	
書き込み保護	なし	あり	なし	あり	なし	あり	なし	あり	なし	あり	なし	あり	なし	あり	なし	あり	なし	あり	なし	あり
BS セグメント / コード保護ヒューズの消去	GS/SS/BS/VS セグメントおよび GS/SS/BS コード保護ヒューズを消去																			
GS セグメント / コード保護ヒューズの消去	GS/SS セグメントおよび GS/SS コード保護ヒューズを消去 ブートセグメントが定義されていない場合は VS を消去																			
GS セグメント / コード保護ヒューズの消去	GS セグメントおよび GS コード保護ヒューズを消去 ブートセグメントが定義されていない場合は VS を消去																			
GS セグメントの消去	GS セグメントのみを消去																			
コンフィグレーション レジスタのプログラム	可																			

Note 1: プログラムフロー変更 (PFC) は、PC に通常の自動インクリメントされた値ではなく、他の新しい値が読み込まれる事と定義します。JUMP、CALL、RETURN、RETFIE、計算型ジャンプ等の命令があります。

2: PFC のターゲットとして許されるのはセグメント内の最初の 32 命令位置のみです。

3: VS セグメント内では実行が許可されていないため、このような条件はあり得ません。

4: IVT および AIVT セグメントへの PFC 動作 (すなわち分岐、呼び出し等) は可能です。ただし、左記のセグメントからこれを試みると、即座に不正アドレストラップが発生します (フロー変更先にアドレス 0x000000 のリセットベクタを指定した場合を除く)。

5: ベクタフロー変更 (VFC) は、PC に割り込みまたはトラップのベクタアドレスが読み込まれた場合と定義されます。

6: この動作は、より高いセキュリティ権限のセグメントが定義されていない場合にのみ許可されます。

7: TBLRD または DS 読み出しは実行可能です。ただし、許可されていない場合はオール「0」が返されます。

8: この動作が実行可能なデバイスは、dsPIC33FJXXXGPX06/X08/X10、dsPIC33FJXXXMCX06/X08/X10、PIC24HJXXXGPX06/X08/X10 のみです。その他の全てのデバイスでは、この動作は禁止されています。

23.11 プログラムフローに関するルール

プログラムフローとは、プログラムメモリ内のプログラム命令の実行シーケンスを意味します。通常、命令はプログラム カウンタ (PC) のインクリメントに従って順次実行されます。コード保護を実装している場合、プログラムフローは権限レベルに従います。つまり、コード保護されたメモリから実行されるプログラムのフローは、高いセキュリティ レベルのセグメントから低いセグメントに流れる事はあっても、その逆に進む事はできません。例えば、セキュア セグメントから実行されるプログラムフローは、汎用セグメントへと流れる場合はあっても、ブートセグメントに向かう事はありません。

プログラムフロー変更 (PFC) は、呼び出し、ジャンプ、計算型ジャンプ、リターン、サブルーチンからのリターン、その他の分岐命令の結果、PC が再読み込みされると発生します。PFC を使うと、プログラム フローを変更できます。標準 PFC によるプログラムの分岐先は、同じセグメント内に限られます。これに対して制限付き PFC では、より高いセキュリティ セグメントの特殊セグメント アクセス領域に分岐できます。

ベクタフロー変更 (VFC) は、PC に割り込みまたはトラップベクタが再読み込みされると発生します。

意図しない位置にある保護コードにプログラムがジャンプした場合、アルゴリズム検出によってコードが漏洩する恐れがあります。権限階層に違反する PFC/VFC 動作が制限されているのはこの理由からです。

図 23-8: プログラムフローのルール

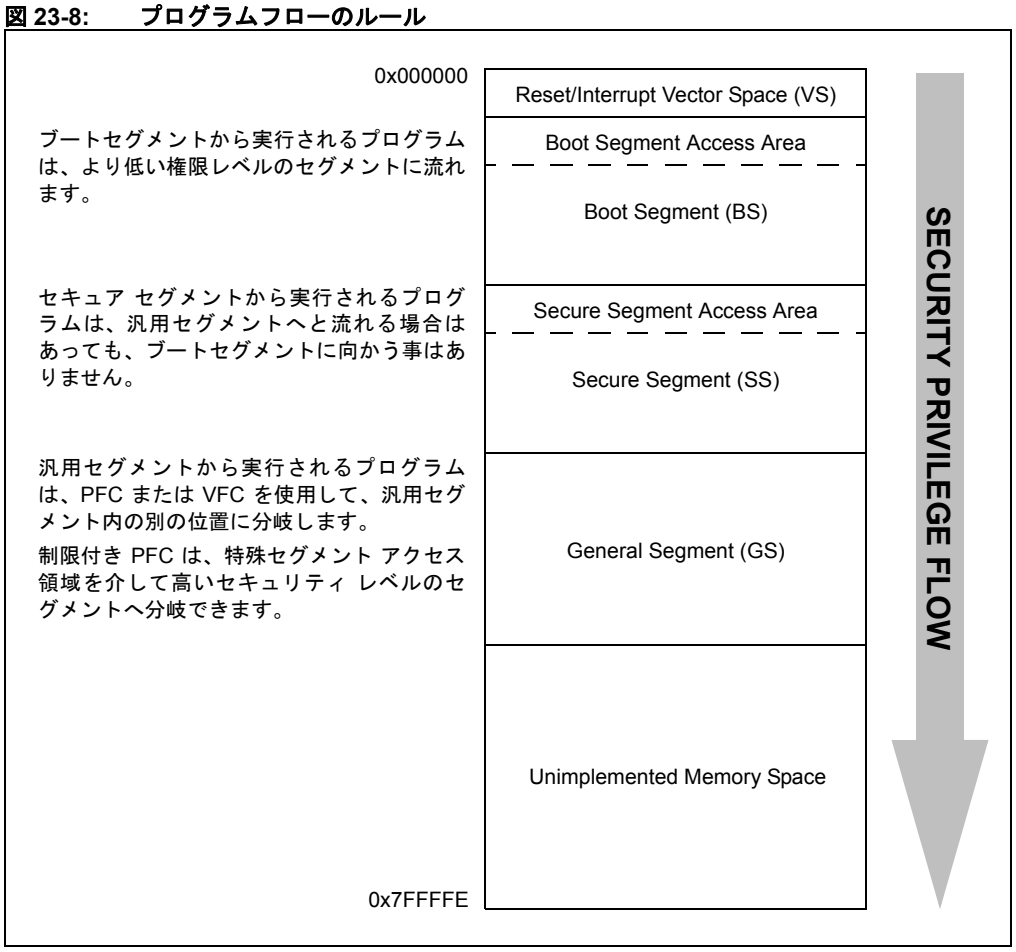


表 23-18 は、プログラムメモリ セグメント間で実行可能な動作の概要です。空欄は、読み出し / 書き込み / 消去 /PFC 動作が実行できない事を示します。ブートセグメントまたはセキュア セグメントで高セキュリティ レベルを実装している場合、低い権限レベルからの PFC はセグメント アクセス領域に限定する必要があります。セグメント アクセス領域以外への PFC は、全てセキュリティ リセットを発生させます。詳細は、**23.11.3「プログラムフローのエラー」**を参照してください。

表 23-18: プログラムメモリ セグメント間で可能な動作

コード実行元		動作ターゲット						
		ブートセグメント セキュリティ レベル		セキュア セグメント セキュリティ レベル		汎用セグメント セキュリティ レベル		
		標準	高	標準	高	標準	高	なし
BS	標準	R, P, PFC	—	R, P, PFC	PFC*	R, P, PFC	PFC	R, P, PFC
	高	—	R, P, PFC	R, P, PFC	PFC*	R, P, PFC	PFC	R, P, PFC
SS	標準	PFC	PFC*	R, P, PFC	—	R, P, PFC	PFC	R, P, PFC
	高	PFC	PFC*	—	R, P, PFC	R, P, PFC	PFC	R, P, PFC
GS	標準	PFC	PFC*	PFC	PFC*	R, P, PFC	—	—
	高	PFC	PFC*	PFC	PFC*	—	R, P, PFC	—
	なし	PFC	PFC*	PFC	PFC*	—	—	R, P, PFC

凡例： R – 読み出し
P – プログラム (書き込み)/ 消去
PFC – セグメント内の任意の位置に分岐できるプログラムフロー変更
PFC* – 制限付きプログラムフロー変更(セグメント アクセス領域への分岐のみ可能)

23.11.1 フロー変更

セグメント内の PFC に制限はありません。通常、セグメント間の PFC と VFC も下記の場合を除いて制限される事はありません。

- ブートセグメントとセキュア セグメント内のコード実行の整合性を確保するため、ユーザはこれらのセグメントへのプログラムフローを制限する必要があります。プログラムフローを制限する事で、分岐先をセグメント アクセス領域に限定する事ができます。セグメント アクセス領域はブートセグメントまたはセキュア セグメントのコード空間の最初の 32 命令位置です。
- ブートセグメントまたはセキュア セグメントのセキュリティ レベルを「高」に設定した場合、より優先度が低いセグメントからの PFC のターゲットは、このアクセス領域内である必要があります。
- ブートセグメントまたはセキュア セグメントのセキュリティ レベルが「高」に設定されている状態で VFC が発生した場合、そのターゲットはセグメント アクセス領域内である必要があります。

ブートセグメントまたはセキュア セグメント コード空間内のコードの所有者は、このアクセス領域からアプリケーション コードの指定部分以外に分岐しないようにする事で、アルゴリズムの漏洩を防止できます。

23.11.2 リセット命令

通常のデバイスリセットでは、PC が 0x000000 にリセットされ、その位置からプログラム実行が開始されます。コード開始位置への分岐命令は、0x000000 ～ 0x000002 の位置に配置する必要があります。

リセットベクタの命令は、ブートセグメントまたはセキュア セグメントが高セキュリティでない限り任意の位置に分岐できます。高セキュリティの場合、指定済みセグメント アクセス領域をターゲットとする必要があります。通常、RESET 命令のターゲットは汎用セグメントのコード空間です。

23.11.3 プログラムフローのエラー

制限されたメモリ位置をターゲットとしたPFCまたはVFCはセキュリティ リセットを発生させます。デバイスがリセットされると同時に、不正な動作を示すIOPUWR (RCON<14>) ステータスビットがセットされます。

このセキュリティ リセットの他にも、全てのデバイスはプログラムフロー チェック機能を内蔵しています。

プログラムフロー変更またはベクタフロー変更が未実装のプログラムメモリ空間をターゲットにした場合、アドレス エラー トラップが発生します。

リセット位置にある命令を除き、ベクタセグメントからのコード実行は禁止です。この場合には、アドレス エラー トラップが発生します。

23.11.4 ブート後リセットのターゲット変更

リセット動作は、リセットが発生した時点にデバイスがどのセグメントで動作していたかには依存しません。ブートセグメントを割り当てた場合、コード プロービングを防止するためにリセットベクタが保護されます。ブートセグメントが存在する場合、リセットベクタを含むベクタ空間はブートセグメントの一部となり、ブートセグメントのルールによって保護されます。ブートセグメントが存在しない場合、リセットベクタを含むベクタ空間は汎用セグメントの一部となり、汎用セグメントによる変更が可能です。

例えば、ブートローダを含むブートセクタがあるとします。リセットが発生すると、デバイスはブートローダ内の位置にリセットします。ブートローダが実行され、汎用セグメント内にユーザコードを読み込みます。その動作が完了後、ブートローダがリセットベクタ命令を書き換え、ユーザコードを指すようにできます。この場合、次回リセットではユーザコードにリセットしてしまいます。しかし、ユーザコードはリセットベクタ命令を書き換える事はできません。

23.12 割り込みに関するルール

23.12.1 セキュアモードにおける割り込みとトラップ

割り込み処理は、以下に示す理由で制限されます。

- 割り込みからのリターンは、(スタック内のリターンアドレスを変えてしまい) 意図したプログラムフローを乱す一因になる
- セキュアなコードは、割り込みに応答する前に機密性の高い情報を消去する必要がある

23.12.1.1 BS および SS の割り込みベクタ

ブートセグメントまたはセキュア セグメント内でプログラム実行中に割り込みが発生した場合、プロセッサはブートセグメント特殊割り込みベクタの位置 (BS + 0x20) または、セキュアセグメント特殊割り込みベクタの位置 (SS + 0x20) から割り込みベクタを取得します。

Note: 特殊割り込みベクタは、ブートセグメントとセキュア セグメントに 1 つずつ存在します。これらのセグメントのいずれかでコード実行中に割り込みが発生すると、プロセッサはそのセグメントの特殊割り込みベクタに分岐します。ユーザは保護セグメント内の特殊 ISR を使って機密データを秘匿し、その後 INTTREG SFR を読み出して実際の ISR へ分岐できます。

23.12.1.2 割り込みおよびトラップの処理シーケンス

割り込みおよびトラップの処理シーケンスは以下の通りです。

1. ブートセグメント等の保護されたセグメントでコードを実行中に割り込みまたはトラップが発生します。
2. リターンアドレスがスタックにプッシュされます。
3. 通常の割り込みベクタの代わりに、メモリ位置 (BS + 0x20) の内容が PC に読み込まれます。
4. (BS + 0x20) が指すアドレスに格納された特殊 ISR が実行されます。
5. W レジスタ内の機密情報は、ブートセグメントまたはセキュア セグメント内のセキュア RAM 領域に格納されます。
6. 実際のリターンアドレスをスタックから取得し、セキュア RAM に保存します。
7. 実際のリターンアドレスを新しいリターンアドレスで置き換えます (例 : BS + 0x30)。BS + 0x30 は、BS から BS + 0x3E のアドレス範囲にあります。この範囲はブートセグメント外からの PFC が許可されている事に注意してください。
8. INTTREG SFR を読み出し、どの割り込みベクタにジャンプすべきかを判断します。
9. ベクタテーブルから割り込みベクタを読み出し、間接ジャンプを実行します。
10. ユーザ ISR が開始します。
11. ユーザコードが実行されます。
12. 割り込みからリターンします (BS + 0x30 に戻ります)。
13. セキュア RAM 領域から実際のリターンアドレスを読み出します。
14. W レジスタの値を復元します。
15. ブートセグメントに戻るために間接ジャンプを実行します。

表 23-19: 通常のユーザモードにおけるベクタ動作

ベクタ動作	通常のユーザモードにおける結果
BS 内コード実行中のハードウェア割り込み	BS 特殊 ISR ベクタ位置 (BS + 0x20) からベクタを取得
SS 内コード実行中のハードウェア割り込み	SS 特殊 ISR ベクタ位置 (SS + 0x20) からベクタを取得
GS 内コード実行中のハードウェア割り込み	通常の ISR ベクタ位置からベクタを取得
BS 内コード実行中のソフトウェア割り込みおよびトラップ	BS 特殊 ISR ベクタ位置 (BS + 0x20) からベクタを取得
SS 内コード実行中のソフトウェア割り込みおよびトラップ	SS 特殊 ISR ベクタ位置 (SS + 0x20) からベクタを取得
GS 内コード実行中のソフトウェア割り込みおよびトラップ	通常の ISR ベクタ位置からベクタを取得

23.13 RAM データアクセスのルール

23.13.1 セグメント RAM の使用法

ブートセグメントまたはセキュア セグメントに保護 RAM 空間を割り当てた場合、その RAM はセグメント外で実行されるコードからはアクセスできません。例えば、セキュア セグメントまたは汎用セグメントで実行中のコードは、ブートセグメントによって保護された RAM にアクセスできません。

保護された RAM 位置に対する不正な読み出し命令が発行された場合、読み出しは実行されますが、結果の書き込みが無効化されます。例えば、「`mov ssram, w0`」という命令は SSRAM 位置からの読み出しを実行しても、`w0` への書き込みは実行しません。ALU からの結果は全てゼロになり、書き込み動作は発生しません。

ブート RAM セグメント内の保護された RAM 位置に対する不正読み出しが実行されると、IR_BSR (BSRAM<1>) ビットがセットされます。このビットは、デバイスリセットが発生するか、ブートセグメント内で実行中のコードによって BSRAM レジスタが読み出されるまでセットされたままです。

同様に、セキュア RAM セグメントからの不正な読み出しは、IR_SSR (SSRAM<1>) ビットをセットします。このビットは、デバイスリセットが発生するか、セキュア セグメント内で実行中のコードによって SSRAM レジスタが読み出されるまでセットされたままです。

保護された RAM 位置に不正な書き込みを実行すると、保護位置には「`0`」が書き込まれます。

保護された RAM 位置への不正な書き込みは、不正書き込みステータスビット IW_BSR (BSRAM<2>) または IW_SSR (SSRAM<2>) のいずれかをセットします。これらのビットは、デバイスリセットが発生するか、ブートセグメント内またはセキュア セグメント内で実行中のコードによって BSRAM または SSRAM レジスタが読み出されるまでセットされたままです。

23.13.2 スタック割り当て

ユーザは RAM 内の任意の位置にスタック空間を割り当てる事ができます。ただし、ブートセグメントとセキュア セグメントの両方からアクセスできる領域は汎用セグメント RAM 内に限られます。従って、スタックは汎用セグメント RAM 領域に割り当てる必要があります。

スタックが偶発的に「BSRAM」または「SSRAM」領域にはみ出した場合、プッシュによる書き込みおよびポップによる読み出しは、上述の不正 RAM アクセスと見なされます。

このような状況は、スタックポインタ リミット レジスタ (SPLIM) を使って、そのレジスタの内容を適切なアドレスに設定する事で予防できます。

23.13.3 レジスタダンプに対する保護

デバイスは全てのリセットで W レジスタを初期化します。データ RAM は初期化されないため、リセットが発生してもデータ RAM 内に有効なデータが残る可能性があります。このため、RAM の内容に対するセキュリティを確保する必要があります。

23.14 セキュリティ機能とデバイスの動作モード

セキュリティ機能は、デバイスの動作モードに依存します。各デバイスは、以下のいずれかのモードで動作します。

- 実行時自己プログラミング (RTSP) モード (通常のデバイス動作) では、アプリケーション コードが実行され、アプリケーション コードが自己プログラミングを開始できます。
- インサーキット シリアル プログラミング (ICSP™) モードは、デバイスを消去、プログラム、ベリファイするローレベル ネイティブ プログラミング機能を提供します。このモードのデバイスは PRO MATE® 3 または MPLAB® ICD 2 等のデバイス プログラマによって制御されます。

23.14.1 RTSP モードによるデバイス プログラミングのルール

デバイスの自己プログラミングでは、はじめにコードの一部をクリアするために消去コマンドが実行されます。続いて、書き込みラッチに新しいコードまたはデータが読み込まれ、最後にプログラミング コマンドによって書き込みラッチの内容がフラッシュアレイにプログラムされます。消去またはプログラミング コマンドは、デバイス固有の NVMCON SFR によって設定します。NVMOP ビットフィールドによって特定の機能を選択し、ERASE ビットによってプログラミングまたは消去のどちらの機能を使用するかを選択します。プログラミング動作は NVMCON レジスタの WR ビットによって開始されます。従って、コードの整合性を保護するため、WR ビットのセットによって起動する動作は制限されます。

23.14.1.1 コード行またはコードページの消去とプログラミング

フラッシュアレイの実装方法に応じて、NVMOP ビットはプログラム フラッシュ アレイのページ消去またはページ プログラミングを設定します。

- セグメントの書き込み保護を有効にすると、そのセグメントでは消去またはプログラミングが一切実行されません。
- あるセグメント内で実行中のコードは、同一セグメントの一部を消去またはプログラムできます。
- 優先度の高いセグメントで実行中のコードは、より優先度の低いセグメントの一部を消去またはプログラムできます。ただし、低優先度のセグメントのセキュリティ レベルが「高」に設定されていない場合に限ります。
- ベクタ空間がブートセグメントまたは汎用セグメントに設定された高セキュリティ属性を継承している場合、いかなるセグメントからも消去またはプログラムできません。ブートセグメントが定義されている場合、ベクタ空間を消去またはプログラムできるのはブートセグメントだけです。ブートセグメントが定義されていない場合、全てのセグメントがベクタ空間を消去またはプログラムできます。

23.14.1.2 セグメントの消去とコード保護のクリア

1 回の動作でプログラム フラッシュの全セグメントまたはより低優先度のセグメントの全てを消去し、さらにコード保護に関連するコンフィグレーション ビットをクリアする数種類の NVMOP コマンドが用意されています。これは、セグメントに対するコード保護を解除する唯一の方法です。これらのコマンドは、任意のセグメントで実行中のコードで使用できます。ただし、セグメント RAM の内容は一切消去されません。

23.14.2 ICSP モードによるデバイス プログラミングのルール

デバイスをデバイス プログラマに接続している時に可能な動作は、デバイスのコードメモリの消去、プログラミング、ベリファイだけです。

- デバイス プログラマは、デバイスを消去してコード保護をクリアするセグメント消去コマンドを使用します。
- コード保護が選択されている場合、そのレベルの高低にかかわらずプログラミング コマンドは無視されます。プログラムするには、ブートセグメントまたはセキュア セグメントを指定せず、汎用セグメントにコード保護を設定しない事が必要です。
- いずれのレベルであれコード保護が設定されているデバイスはベリファイできません。コード保護されたデバイスをベリファイしようとすると「0」が読み出されます。

デバイスのプログラミングが完了すると、コード保護レベルを有効化するコンフィグレーション ビットが書き込まれます。この動作以降にデバイスのコードを変更するには、コード自体で自己プログラミングを実行するか、再度消去とコード保護のクリアを実行する必要があります。

23.15 代表的なデバイスのブートロード手順

コード保護を使用するデバイスのブートロードの代表的なシナリオは、フィールド アップグレードです。この場合、デバイスはブートセグメントと汎用セグメントの 2 つを使用します。汎用セグメントはアプリケーションを格納しています。ブートセグメントは保護されたブートローダを格納しています。両方のセグメントにはセキュリティ レベル「高」が設定されています。

システムリセットが発生すると、デバイスは汎用セグメント内のアプリケーションにジャンプします。

フィールドで稼働しているシステムに技術者が再プログラミング ツールを接続します。アプリケーションはこの接続を認識し、ブートセグメントのアクセス領域に分岐します。この分岐には高度なセキュリティが設定されているため、変更しようとするとデバイスリセットが発生します。

ブートセグメントには、ツールとの間の暗号化通信を可能とするコードが含まれます。暗号化キーはブートセグメント コードに格納されており、外からアクセスできないため安全です。ブートローダを最初にシステムにプログラムする時にシリアル プログラミングを使用すれば、暗号化キーはそのシステム固有のものする事ができ、暗号化通信の強度をさらに高める事ができます。

ブートローダが外部のプログラミング ツールとの間に有効な通信が確立された事を確認すると、汎用セグメント内のコードを消去し、汎用セグメントのコード保護をクリアします。

さらにブートローダは暗号化された更新コードをツールから受信し、復号して汎用セグメントにプログラムします。

ブートローダの動作中は割り込みやトラップによって中断されることはありません。割り込みまたはトラップは、ブートローダ内の安全な位置にジャンプするためです。

ブートローダは動作を完了すると汎用セグメントを再度保護するためにコンフィグレーション ビットをプログラムし、必要に応じてベクタを更新したのち、アプリケーションに戻ります。

23.16 保護されたサードパーティ製アルゴリズムの代表的な実装方法

ここでは、システム インテグレータがサードパーティ ベンダからアルゴリズムを購入するシナリオを考えます。この場合、システム インテグレータはサードパーティ アルゴリズムの障害から自社のシステムコードを保護する必要があります。一方、アルゴリズムのサードパーティ ベンダはシステム インテグレータが所有する企業を経由して自社コードが流出する事を望みません。

通常これはサードパーティ ベンダの信頼の問題です。なぜなら、インテグレータが自社のシステムコードにリンクし、デバイスにプログラムするネイティブコードは、サードパーティ ベンダが提供しなくてはならないためです。

サードパーティ ベンダがコードを暗号化して提供でき、デバイスはこのコードを他のコードと分離して扱う事ができれば、サードパーティ ベンダはコードを開示する必要がありません。

このシナリオの場合、デバイスはブート、セキュア、汎用の各セグメントを割り当てます。システム インテグレータのコードは汎用セグメントに格納し、上述と同様のブートローダをブートセグメントに格納します。

システム インテグレータはブートローダとアプリケーション コードをデバイスにプログラムします。

サードパーティ ベンダが提供する特殊なローダはセキュア セグメントにプログラムします。このローダは、サードパーティ ベンダからシステム インテグレータに提供されるキーを使用して、サードパーティ アルゴリズムを復号し、デバイスにプログラムします。セキュア セグメントが保護されると、ブートセグメントまたは汎用セグメントにプログラムされたシステム インテグレータのコードからは、このセグメント内のアルゴリズムにアクセスできなくなります。サードパーティのコードには、セキュア セグメントのアクセス領域に対する呼び出しによってのみアクセスできます。

保護対象のアルゴリズムは、機密データパラメータを保護 RAM 領域内に保持する必要があります。アルゴリズムが動作を完了して他のセグメント内のコードに戻る前には、RAM 領域を「洗浄」する必要があります。

23.17 設計のヒント

質問 1: **基本コード保護設定のデバイスでブートローダを使用できますか。**

回答 : 基本コード保護設定のデバイスには、汎用セグメントしかない事を思い出してください。セグメントが 1 つしか存在しないため、そこに格納されているかもしれないブートローダ自体を消去せずにそのセグメントを消去し、コード保護をクリアする事は不可能です。

このためブートの選択肢は限られます。ただし、ブートが全く不可能というわけではありません。この場合、ブートローダは「1 セグメントよりも小さな」パーティションを消去し、再プログラムする必要があります。また、ローダによる汎用セグメントの書き込み保護設定も不可です。読み込まれたコードをブートローダ自体に起因する漏洩から保護する事もできません。

質問 2: **システムがまずコードの一部を読み込み、残りを後で読み込む事はできますか。**

回答 : セグメントに対して書き込み保護および高セキュリティのいずれも設定されていなければ「インクリメンタル」読み込みは可能です。また、高セキュリティのセグメントであっても、ローダがそのセグメントに格納されていればインクリメンタル読み込みを実行できます。しかし、セグメントに書き込み保護が設定された後は、セグメント消去コマンドによってセグメント全体を消去してコード保護をクリアするまでコードは変更できません。

割り込みベクタ用のジャンプテーブルを保護対象外のセグメントに設定し、割り込みベクタを変更する事でジャンプテーブルを更新する方法もあります。この方法であれば、ブートセグメントを書き込み保護にする事ができます。

23.18 関連文書

マニュアルの本セクションに関連する文書の一覧を示します。一部の文書はdsPIC33F製品ファミリ向けではありません。ただし概念は共通しており、変更が必要な場合や制限事項が存在する場合があるものの適用は可能です。CodeGuard™ セキュリティに関連する、現在提供中の文書は以下の通りです。

文書タイトル	文書番号
CodeGuard™ Security: Protecting Intellectual Property in Collaborative System Designs	DS70179

Note: dsPIC33F ファミリ向けのその他のアプリケーション ノートとサンプルコードは、マイクロチップ社のウェブサイト (www.microchip.com) でご覧になれます。

23.19 改訂履歴

リビジョン A (2007 年 3 月)

初版発行

リビジョン B (2007 年 5 月)

本書全体の小規模な更新

リビジョン C (2009 年 9 月)

このリビジョンでの変更内容は以下の通りです。

- Notes:
  - 表 23-17 に下記の注釈を追加 この動作が実行可能なデバイスは、dsPIC33FJXXXGPX06/X08/X10、dsPIC33FJXXXMCX06/X08/X10、PIC24HJXXXGPX06/X08/X10 のみです。その他の全てのデバイスでは、この動作は禁止されています。
- 表現および体裁の変更等、本書全体の細部を修正

ISBN: 978-1-60932-872-6