注意 : この日本語版文書は参考資料としてご利用ください。最新情報は必ずオリジナルの英語版をご参照願います。

セクション 5. フラッシュ プログラミング

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。

Note: ファミリ リファレンス マニュアルの本セクションは、デバイス データシートの内容を補足する事を目的としています。本セクションの内容は、dsPIC33F/PIC24H ファミリの一部のデバイスには対応していません。

本書の内容がお客様のご使用になるデバイスに対応しているかどうかは、最新デバイス データシート内の**「フラッシュ プログラムメモリ」**の冒頭に記載している注意書きでご確認ください。

デバイス データシートとファミリ リファレンス マニュアルの各セクションは、マイクロチップ社のウェブサイトからダウンロードできます (http://www.microchip.com)。

5.1 はじめに

本セクションでは、フラッシュ プログラムメモリのプログラミング方法について説明します。dsPIC33F/PIC24Hファミリのデバイスは、ユーザコードの実行用にプログラマブル フラッシュ メモリを内蔵しています。このメモリは下記の 2 つの方法でプログラミングできます。

- ランタイム自己プログラミング (RTSP)
- インサーキット シリアル プログラミング (ICSP™)

本セクションでは、ユーザ割り当てソフトウェアによる RTSP プログラミングについて説明します。

ICSP では、シリアルデータ接続を用いて RTSP よりも高速なプログラミングが可能です。ICSP プロトコルの定義は『dsPIC33F/PIC24H フラッシュ プログラミング仕様書』(DS70152) に記載しています。この仕様書はマイクロチップ社のウェブサイトからダウンロードできます。

5.2 テーブル命令の動作

テーブル命令は、dsPIC33F/PIC24H のプログラムメモリ空間とデータメモリ空間の間でデータを転送するための手段を提供します。本セクションには、フラッシュ プログラムメモリのプログラミングで使用するテーブル命令の概要を記載します。基本的に下記の 4 つのテーブル命令を使用します。

- TBLRDL: テーブル読み出し LOW
- TBLRDH: テーブル読み出し HIGH
- TBLWTL: テーブル書き込み LOW
- TBLWTH: テーブル書き込み HIGH

TBLRDL 命令は、プログラムメモリ空間内の 16 ビット (<15:0>) からの読み出しに使用します。TBLWTL 命令は、プログラムメモリ空間内の 16 ビット (<15:0>) への書き込みに使用します。TBLRDL と TBLWTL は、ワードモードまたはバイトモードでプログラムメモリへアクセスできます。

TBLRDH および TBLWTH 命令は、プログラムメモリ空間内の 8 ビット (<23:16>) の読み書きに使用します。TBLRDH と TBLWTH は、ワードモードまたはバイトモードでプログラムメモリへアクセスできます。プログラムメモリは 24 ビット幅しか持たないため、上位バイトは実在しません。しかし TBLRDH および TBLWTH 命令は、この実在しない上位バイトをアドレスできます。このバイトを「幽霊バイト」(phantom byte) と呼びます。この上位バイトの読み出しは常に 0x00 を返します。このバイトへの書き込みは何も効果を持ちません。

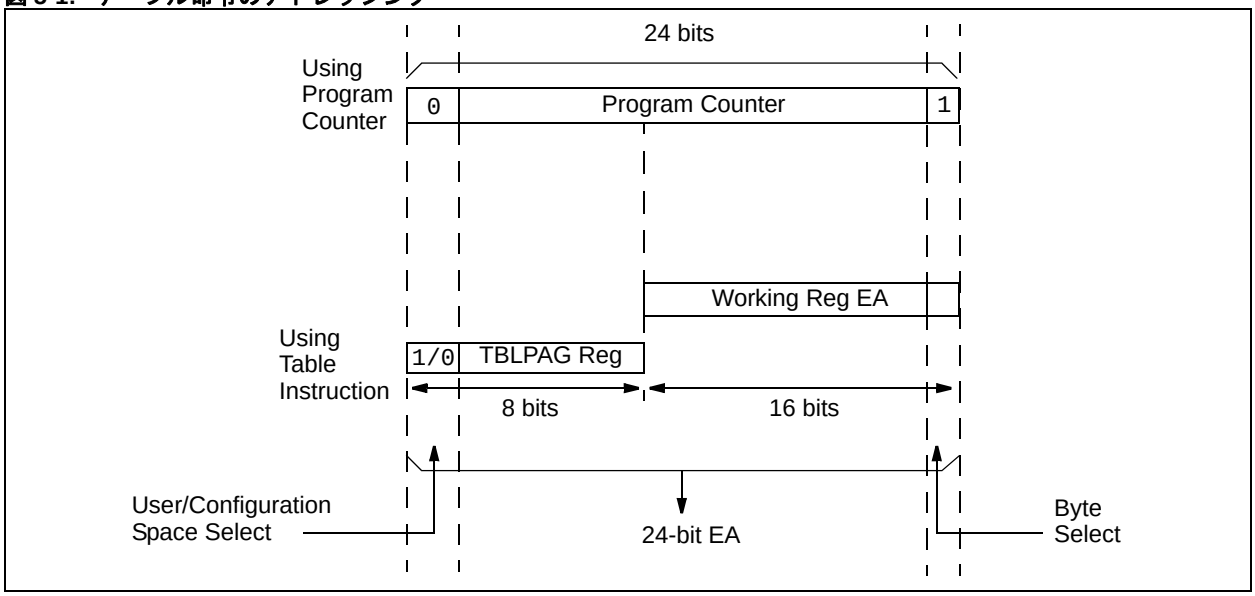
24 ビットのプログラムメモリは、同一アドレスレンジを共有する 2 つの隣り合った 16 ビット空間として見なされます。TBLRDL および TBLWTL 命令は「下位 (LOW)」プログラムメモリ空間 (PM<15:0>) へアクセスし、TBLRDH および TBLWTH 命令は「上位 (HIGH)」プログラムメモリ空間 (PM<31:16>) へアクセスします。PM<31:24> に対する読み書きは、実在しない「幽霊」(phantom) バイトへアクセスします。いずれのテーブル命令でも、バイトモードで使用する場合には、テーブルアドレスの最下位ビット (LSb) をバイト選択ビットとして使用します。この LSb によって、上位と下位のどちらのプログラムメモリ空間へアクセスするのかが決まります。

図 5-1 に、テーブル命令を用いたプログラムメモリのアドレッシング方法を示します。24 ビットのプログラムメモリ アドレスの形成には、TBLPAG レジスタの 8 ビット (<7:0>) と、テーブル命令内で指定したワーキング レジスタからの有効アドレス (EA) を使用します。図 5-1 には、参考のために 24 ビットのプログラムカウンタも記載しています。プログラムメモリ位置の選択には、上記のように生成した 24 ビット有効アドレス (EA) の上位 23 ビットを使用します。

バイトモードのテーブル命令では、16 ビット プログラムメモリ ワード内のどちらのバイトをアドレスするのかを選択するために、ワーキング レジスタ有効アドレスの LSb を使用します。この LSb が「1」であれば、上位バイト (ビット <15:8>) を選択します。この LSb が「0」であれば、下位バイト (ビット <7:0>) を選択します。ワードモードのテーブル命令では、ワーキング レジスタ有効アドレスの LSb を無視します。

テーブル命令は、プログラムメモリ アドレス以外に、ワーキング レジスタ (またはメモリ位置を指すワーキング レジスタ ポインタ) も指定します。これにより、プログラムメモリへの書き込みデータのソース、またはプログラムメモリからのデータの読み出し先を指定します。バイトモードのテーブル書き込み動作では、ソース ワーキング レジスタの上位バイト (ビット <15:8>) を無視します。

図 5-1: テーブル命令のアドレッシング



5.2.1 テーブル読み出し命令の使用方法

テーブル読み出しには下記の 2 ステップが必要です。

- TBLPAG レジスタと、いずれか 1 つのワーキング レジスタを使用してアドレスポインタを形成する
- アドレス位置のプログラムメモリ内容を読み出す

5.2.1.1 ワードモードの読み出し

テーブル命令をワードモードで使用してプログラムメモリ内の 1 ワードを読み出すためのサンプルコードを例 5-1 に示します。

例 5-1: ワードモードの読み出し

```
; Set up the address pointer to program space
    MOV         #tblpage(PROG_ADDR),W0        ; get table page value
    MOV         W0,TBLPAG                     ; load TBLPAG register
    MOV         #tbloffset(PROG_ADDR),W0      ; load address LS word
; Read the program memory location
    TBLRDH      [W0],W3                       ; Read high byte to W3
    TBLRDL      [W0],W4                       ; Read low word to W4
```

5.2.1.2 バイトモードの読み出し

例 5-2 のサンプルコードでは、下位バイト読み出し時のポストインクリメント演算子によって、ワーキング レジスタ内のアドレスを 1 つインクリメントます。これは EA<0> を「1」にセットします。これにより、次の読み出し命令は中央バイトへアクセスします。最後の読み出し命令で再度ポストインクリメントする事によって、W0 は次のプログラムメモリ位置 (偶数アドレス) を指します。

Note: tblpage( ) と tbloffset( ) 疑似命令は、dsPIC33F/PIC24H 向けのマイクロチップ社製アセンブラによって提供されます。これらの疑似命令は、1 つのプログラム メモリ アドレス値から、テーブル命令向けに適切な TBLPAG 値とワーキング レジスタ値を選択します。詳細は『PIC24 MCU および dsPIC®DSC 向け MPLAB® アセンブラ、リンカ、ユーティリティ ユーザガイド』(DS51317) を参照してください。

例 5-2: バイトモードの読み出し

```
; Set up the address pointer to program space
    MOV         #tblpage(PROG_ADDR),W0        ; get table page value
    MOV         W0,TBLPAG                     ; load TBLPAG register
    MOV         #tbloffset(PROG_ADDR),W0      ; load address LS word
; Read the program memory location
    TBLRDH.B    [W0],W3                       ; Read high byte to W3
    TBLRDL.B    [W0++],W4                     ; Read low byte to W4
    TBLRDL.B    [W0++],W5                     ; Read middle byte to W5
```

5.2.1.3 テーブル書き込みホールドバッファ

Note: 一部の dsPIC33F/PIC24H デバイスは複数のホールドバッファを備えていません。詳細は各デバイスのデータシートに記載されている**「フラッシュ プログラムメモリ」**セクションを参照してください。

テーブル書き込み命令は不揮発性プログラムメモリへ直接書き込むのでなく、ホールドバッファに書き込みデータを格納します。これらのホールドバッファはメモリ空間には割り当てられず、テーブル書き込み命令だけがアクセスできます。全てのホールドバッファへデータを格納した後に、特殊な命令シーケンスを実行する事によってメモリへのプログラミング動作を開始します。

dsPIC33F/PIC24H のフラッシュ プログラムメモリは、複数のページ (512 ワード) と行 (64 命令ワード) に分割されます。

dsPIC33F/PIC24H ファミリのデバイスは、1 行のメモリ プログラミングに対して同時に 64 個のホールドレジスタをサポートします。メモリロジックは、テーブル書き込み命令内のアドレス値に基づいて、書き込みバッファのどのセットを読み込むのかを決定します。

5.2.1.4 ワードモードの書き込み

例 5-3 のサンプルコードは、ワードモードによる単一プログラムメモリ ラッチ位置の書き込みに使用できます。

例 5-3: ワードモードでの単一プログラムメモリ ラッチ位置の書き込み

```
; Setup the address pointer to program space
    MOV         #tblpage(PROG_ADDR),W0    ; get table page value
    MOV         W0,TBLPAG                 ; load TBLPAG register
    MOV         #tbloffset(PROG_ADDR),W0  ; load address LS word
; Load write data into W registers
    MOV         #PROG_LOW_WORD,W2
    MOV         #PROG_HI_BYTE,W3
; Perform the table writes to load the latch
    TBLWTL      W2,[W0]
    TBLWTH      W3,[W0++]
```

例 5-3 では、W3 の上位バイトの内容は重要ではありません。このデータは幽霊 (phantom) バイト位置へ書き込まれます。2 番目の TBLWTH 命令の後に、次のプログラムメモリ位置への書き込みに備えて W0 を 2 つポストインクリメントします。複数ホールドバッファを持たないデバイスでは、次のプログラミング位置への書き込みを行う前に、フラッシュメモリへの プログラミング サイクルを実行する必要があります。

5.2.1.5 バイトモードの書き込み

例 5-4 のサンプルコードは、バイトモードでの単一プログラムメモリ ラッチ位置の書き込みに使用できます。

例 5-4: バイトモードでの単一プログラムメモリ ラッチ位置の書き込み

```
; Setup the address pointer to program space
    MOV        #tblpage(PROG_ADDR),W0    ; get table page value
    MOV        W0,TBLPAG                 ; load TBLPAG register
    MOV        #tbloffset(PROG_ADDR),W0  ; load address LS word
; Load data into working registers
    MOV        #LOW_BYTE,W2
    MOV        #MID_BYTE,W3
    MOV        #HIGH_BYTE,W4
; Write data to the latch
    TBLWTH.B   W4,[W0]                   ; write high byte
    TBLWTL.B   W2,[W0++]                 ; write low byte
    TBLWTL.B   W3,[W0++]                 ; write middle byte
```

例 5-4 のサンプルコードでは、下位バイト書き込み時のポストインクリメント演算子によって、ワーキング レジスタ (W0) 内のアドレスを 1 つインクリメントします。これは EA<0> を「1」にセットします。これにより、次の書き込み命令は中央バイトへアクセスします。最後の書き込み命令で再度ポストインクリメントする事によって、W0 は次のプログラムメモリ位置 (偶数アドレス) を指します。複数ホールドバッファを持たないデバイスでは、次のプログラミング位置へのき書き込みを行う前に、フラッシュメモリへのプログラミング サイクルを実行する必要があります。

5.3 制御レジスタ

プログラム フラッシュメモリの消去とプログラミングには2つの特殊機能レジスタ (SFR) を使用します (NVMCON と NVMKEY)。

フラッシュメモリ制御レジスタ (NVMCON) は、消去するメモリセグメントの選択、消去サイクル、ヒューズ / フラッシュ動作、ワードまたは行プログラミング、書き込みステータス、バルク消去、プログラミング サイクルの開始を制御します。複数ホールドバッファを持たないデバイスでは、ワード プログラミングだけが可能です。

不揮発性メモリ キーレジスタ (NVMKEY) は、書き込み専用レジスタであり、予期せぬフラッシュ動作からの保護に使用します。プログラミングまたは消去を開始するには、ユーザが NVMKEY レジスタへ 0x55 と 0xAA を連続して書き込む必要があります。

5.3.1 NVMCON レジスタ

NVMCON レジスタは、フラッシュメモリのプログラミング / 消去に使用する重要レジスタです。このレジスタの設定により、消去とプログラミングのどちらを行うのかを選択し、選択した動作を開始する事ができます。

NVMCON レジスタをレジスタ 5-1 に示します。NVMCON レジスタの下位バイトは、実行する NVM 動作のタイプを設定します。

5.3.2 NVMKEY レジスタ

NVMKEY レジスタ (レジスタ 5-2) は書き込み専用レジスタであり、フラッシュメモリの予期せぬ書き込み / 消去を防ぐために使用します。プログラミングまたは消去を開始するには、下記を順番に実行する必要があります。

1. 0x55 を NVMKEY へ書き込む
2. 0XAA を NVMKEY へ書き込む
3. NOP 命令を 2 回実行する

上記を実行した後は、NVMCON レジスタへの書き込みを 1 命令サイクルに 1 回実行できます。多くの場合、プログラミングまたは消去サイクルを開始する前に、ユーザ割り当てアプリケーションが NVMCON レジスタ内の WR ビットをセットする必要があります。キーシーケンス実行中は割り込みを無効にする必要があります。例 5-5 にキーシーケンスを示します。

例 5-5: NVMKEY キーシーケンス

```
    PUSH    SR              ; Disable interrupts, if enabled
    MOV     #0x00E0,W0
    IOR     SR

    MOV     #0x55,W0
    MOV     W0, NVMKEY
    MOV     #0xAA,W0
    MOV     W0, NVMKEY      ; NOP not required
    BSET    NVMCON,#WR      ; Start the program/erase cycle
    NOP
    NOP
    POP     SR              ; Re-enable interrupts
```

詳細は **5.4.2「フラッシュ プログラミング動作」**を参照してください。

レジスタ 5-1: NVMCON: フラッシュメモリ制御レジスタ

R/SO-0(1)	R/W-0(1)	R/W-0(1)	U-0	U-0	U-0	U-0	U-0
WR	WREN	WRERR	—	—	—	—	—
bit 15							bit 8

U-0	R/W-0(1)	U-0	U-0	R/W-0(1)	R/W-0(1)	R/W-0(1)	R/W-0(1)
—	ERASE	—	—	NVMOP<3:0>(2,3)			
bit 7							bit 0

凡例 :	SO = セットのみ可能ビット	
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 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 **WR:** 書き込み制御ビット
1 = フラッシュメモリのプログラミングまたは消去動作を開始 (動作は自動的に実行され、動作終了後にハードウェアがこのビットをクリアします)
0 = プログラミングまたは消去動作は終了 (非アクティブ)

bit 14 **WREN:** 書き込みイネーブルビット
1 = フラッシュ プログラミング / 消去動作は有効
0 = フラッシュ プログラミング / 消去動作を禁止

bit 13 **WRERR:** 書き込みシーケンスのエラーフラグ ビット
1 = プログラミング / 消去シーケンスの不正な試行または終了が発生
(このビットは、WR ビットをセットすると常に自動的にセットされます)
0 = プログラミングまたは消去動作は正常に終了

bit 12-7 **未実装 :**「0」として読み出し

bit 6 **ERASE:** 消去 / プログラミング イネーブルビット
1 = 次の WR コマンドで NVMOP<3:0> が指定する消去動作を実行
0 = 次の WR コマンドで NVMOP<3:0> が指定するプログラミング動作を実行

bit 5-4 **未実装 :**「0」として読み出し

bit 3-0 **NVMOP<3:0>:**NVM 動作選択ビット (2,3)

If ERASE = 1:
1111 = メモリバルク消去動作
1110 = 予約済み
1101 = 一般セグメントと FGS 設定レジスタを消去
1100 = シーケンス セグメントと FSS 設定レジスタを消去
1011 = 予約済み
0011 = 動作なし
0010 = メモリのページ消去動作
0001 = 動作なし
0000 = 設定レジスタの単一バイトを消去

If ERASE = 0:
1111 = 動作なし
1110 = 予約済み
1101 = 動作なし
1100 = 動作なし
1011 = 予約済み
0011 = メモリのワード プログラミング動作
0010 = 動作なし
0001 = メモリの行プログラミング動作
0000 = 設定レジスタの単一バイトをプログラミング

Note 1: これらのビットは、POR によってのみリセット可能です。
2: NVMOP<3:0> の上記以外のビット設定は実装していません。
3: これらのビットの定義はデバイスによって異なります。NVMOP<3:0> ビットの定義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。

レジスタ 5-2: NVMKEY: 不揮発性メモリ キーレジスタ

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

W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0
NVMKEY<7:0>							
bit 7							bit 0

凡例 :	SO = セットのみ可能ビット		
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 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 15-8 **未実装 :** 「0」として読み出し

bit 7-0 **NVMKEY<7:0>:** キーレジスタ (書き込み専用) ビット

5.4 ランタイム自己プログラミング (RTSP)

RTSP は、ユーザ割り当てアプリケーションによるフラッシュ プログラムメモリの書き換えを可能にします。RTSP には、TBLRD (テーブル読み出し) および TBLWT (テーブル書き込み) 命令と、NVM 制御レジスタを使用します。RTSP を使用すると、ユーザ割り当てアプリケーションは 8 行のプログラムメモリ (64 * 8 = 512 命令) を一度に消去し、1 行のプログラムメモリ (64 命令) を一度に書き込む事ができます (ただしデバイスが複数ホールドバッファを備えている場合のみ。これを備えないデバイスでは 1 度に 1 つの命令しか処理できません)。

5.4.1 RTSP 動作

dsPIC33F/PIC24H のフラッシュ プログラムメモリ配列は、複数の行 (1 行 = 64 命令 =192 バイト) によって編成されます。RTSP を使用すると、ユーザ割り当てアプリケーションは 8 行のプログラムメモリ (64*8=512 命令) を一度に消去し、1 行のプログラムメモリ (64 命令) を一度に書き込む事ができます (ただしデバイスが複数のホールドバッファを備えている必要があります。これを備えていないデバイスでは 1 度に 1 つの命令しか処理できません)。8 行の消去ブロックと 1 行の書き込みブロックは、プログラムメモリの開始位置から端が一致するような位置に配置されます (従ってブロック境界はメモリ開始位置から 1536 バイト間隔 (消去) または 192 バイト間隔 (書き込み) に配置されます)。

複数ホールドバッファを備えるデバイスのプログラムメモリは、64 命令のプログラミング データをバッファに格納できます。プログラミング動作を開始する前に、書き込みデータをバッファへ読み込む必要があります。命令ワードは 64 命令間隔の各境界から格納する必要があります。フラッシュメモリを正しくプログラミングするには、直前の TBLWTL または TBLWTH 命令によって書き込まれたアドレスが、これからプログラミングする行に属していなければなりません。複数ホールドバッファを持たないデバイスでは、単一ワードのプログラミングしかできません。

RTSP の基本的シーケンスでは、まずテーブルポインタをセットアップし、次に一連の TBLWTL および TBLWTH 命令を実行してバッファへデータを読み込みます。フラッシュメモリへのプログラミングは、NVMCON レジスタ内の制御ビットをセットする事によって行います。複数ホールドバッファを備えるデバイスでは、バッファへの命令の読み込みに TBLWTL および TBLWTH 命令を 64 回実行する必要があります。複数のホールドバッファを備えていないデバイスでは、TBLWTL および TBLWTH 命令を 1 回だけ実行して 1 つのバッファにだけ読み込み、その都度 NVMCON レジスタ内の制御ビットをセットしてプログラミングを実行します。

全てのテーブル書き込み動作は、バッファへの書き込みだけを行うため、単一ワード書き込み (2 命令サイクル) です。各ワードまたは行をフラッシュ メモリへプログラミングするには、プログラミング サイクルを実行する必要があります。

5.4.2 フラッシュ プログラミング動作

RTSP モードで内蔵フラッシュ プログラムメモリをプログラミングまたは消去するには、プログラミングまたは消去動作が必要です。プログラミングまたは消去動作のタイミング処理は、デバイスが自動的に行います (タイミング情報はデバイスのデータシート参照)。WR ビット (NVMCON<15>) をセットすると動作が始まります。動作が終了すると WR ビットは自動的にクリアされます。

プログラミング動作が終了するまでの間、CPU はストール (ウェイト) します。この間、CPU は命令を一切実行せず、割り込みにも応答しません。プログラミング サイクル中に発生した割り込みは、プログラミング サイクルが終了するまで保留されます。パワーオン リセット (POR) またはブラウンアウト リセット (BOR) が発生すると、プログラミングまたは消去動作は中止されます。この場合ユーザはリセット後にプログラミングまたは消去動作を再度実行する必要があります。プログラミングまたは消去動作中に他のリセット (WDTO、IOPUWR、MCLR、SWR、CM、TRAPR) が発生した場合には、それらのリセット動作は RTSP サイクルが終了するまで保留されます。

5.4.2.1 フラッシュ行プログラミング アルゴリズム

フラッシュ行プログラミングは、複数のホールドバッファを備えたデバイスでのみ可能です。

ユーザ割り当てアプリケーションは、フラッシュ プログラムメモリの 1 行 (64 命令ワード) をプログラミングできます。これを行うには、その行を含むページ (512 命令ワード) を消去する必要があります。

一般的なプロセスは下記の通りです。

1. フラッシュ プログラムメモリの 1 ページ (512 命令ワード) を読み出して、その内容をデータ「イメージ」としてデータ RAM に保存します。RAM イメージは、512 ワード間隔の偶数プログラムメモリ アドレス境界から読み出す必要があります。
2. RAM データイメージを新たなプログラムメモリ データに書き換えます。
3. フラッシュ プログラムメモリ ページを消去します。
   a) NVMCON レジスタをフラッシュ プログラムメモリの 1 ページ消去向けに設定します。
   b) ページ内の任意位置に対してダミーのテーブル書き込みを実行する事によって、消去するページを選択します。
   c) 割り込みを無効にします。
   d) NVMKEY レジスタへキーシーケンスを書き込んで消去を有効にします。
   e) WR ビットをセットします。これにより、消去サイクルが開始されます。
   f) 消去サイクルの間、CPU はストールします。
   g) 消去動作が終了すると、WR ビットがクリアされます。
   h) 割り込みを再び有効にします。
4. テーブル書き込み動作を実行して、1 行の命令ワード (64 命令ワード) を RAM から書き込みバッファへ読み込みます。
5. その行 (64 命令ワード) をフラッシュ プログラムメモリへプログラミングします。
   a) NVMCON レジスタをフラッシュ プログラムメモリの 1 行プログラミング向けに設定します。
   b) 割り込みを無効にします。
   c) NVMKEY レジスタへキーシーケンスを書き込んで、プログラミング サイクルを有効にします。
   d) WR ビットをセットします。これにより、プログラミング サイクルが開始されます。
   e) プログラミング サイクルの間、CPU はストールします。
   f) プログラミング サイクルが終了すると、WR ビットがクリアされます。
   g) 割り込みを再び有効にします。
6. ステップ 4 ～ 6 を繰り返して、プログラムメモリ ページ内の全ての行 (8 行) をプログラミングします。
7. フラッシュ プログラムメモリ内の複数ページをプログラミングする必要がある場合は、ステップ 1 ～ 7 を各ページに対して実行します。

Note 1: RTSP を使用してプログラムメモリを消去する場合、1 度に 512 命令ワードの領域が消去されます。このため、消去サイクルを開始する前に、消去する領域のイメージを汎用 RAM に保存しておく事が重要です。
2: フラッシュ プログラムメモリ内の行またはワードを消去せずに書き込めるのは 2 回までです。

5.4.2.2 フラッシュ単一ワード プログラミング アルゴリズム

このアルゴリズムは、デバイスが複数ホールドバッファを備えているかどうかに関係無く使用できます。

ユーザ割り当てアプリケーションは、フラッシュ プログラムメモリの 1 ワード (1 命令ワード) だけをプログラミングできます。

一般的なプロセスは下記の通りです。

1. テーブル書き込み動作を実行して、1 命令ワードを RAM から書き込みラッチへ読み込みます。
2. その命令ワードをフラッシュ プログラムメモリへプログラミングします。
   a) NVMCON レジスタをフラッシュ プログラムメモリの単一ワード プログラミング向けに設定します。
   b) 割り込みを無効にします。
   c) NVMKEY レジスタへキーシーケンスを書き込んで、プログラミング サイクルを有効にします。

d) WR ビットをセットします。これにより、プログラミング サイクルが開始されます。
e) プログラミング サイクルの間、CPU はストールします。
f) プログラミング サイクルが終了すると、WR ビットがクリアされます。
g) 割り込みを再び有効にします。

3. ステップ 1 と 2 を繰り返して、フラッシュ プログラムメモリに必要なワードをプログラミングします。

5.4.2.3 フラッシュの 1 ページ消去

例 5-6 のサンプルコードは、プログラムメモリの 1 ページ (512 命令) の消去に使用できます。まず NVMCON レジスタをプログラムメモリの 1 ページ消去向けに設定します。次にページ内の任意位置に対するダミーのテーブル書き込みを実行して、消去する行のアドレスを選択します。プログラムメモリは、512 命令ワード間隔の偶数アドレス境界位置で消去する必要があります。従って 1 ページ消去では、テーブル書き込みプログラムメモリ アドレスの下位 10 ビットは効果を持ちません。

キーシーケンスを NVMKEY レジスタへ書き込んでから、WR 制御ビット (NVMCON<15>) をセットして消去動作を開始します。このキーシーケンスは、下記サンプルコードのシーケンスに厳密に従って実行する必要があります。また、途中で割り込みが発生してはなりません。従ってシーケンスを書き込む前に、割り込みを無効にしておく必要があります。

CPU が動作を再開するコード位置には、2 つの NOP 命令を挿入する必要があります。シーケンスの書き込みが終了した後に、必要に応じて割り込みを有効にします。

例 5-6: フラッシュ プログラムメモリの 1 ページ消去

```
; Perform dummy table write to the Page to be erased.
       MOV     #tblpage(PROG_ADDR),W0
       MOV     W0,TBLPAG
       MOV     #tbloffset(PROG_ADDR),W0
       TBLWTL  w0,[w0]
; Setup NVMCON to erase one page of Program Memory
       MOV     #0x4042,W0
       MOV     W0,NVMCON
; Disable interrupts while the KEY sequence is written
       PUSH    SR
       MOV     #0x00E0,W0
       IOR     SR
; Write the KEY Sequence
       MOV     #0x55,W0
       MOV     W0,NVMKEY
       MOV     #0xAA,W0
       MOV     W0,NVMKEY
; Start the erase operation
       BSET    NVMCON,#WR
; Insert two NOPs after the erase cycle (required)
       NOP
       NOP
;Re-enable interrupts, if needed
       POP     SR
```

5.4.2.4 書き込みバッファの読み込み

この動作は、複数のホールドバッファを備えたデバイスにのみ適用可能です。

書き込みバッファは、ユーザ アプリケーションによるテーブル書き込みと行プログラミング シーケンス間のデータ保存手段として使用します。例 5-7 の命令シーケンスは、64 個の書き込みバッファ (64 命令ワード) へのデータ読み込みに使用できます。フラッシュ メモリ 1 行分のプログラミング データを書き込みバッファへ読み込むには、TBLWTL および TBLWTH 命令を 64 回実行する必要があります。

64 命令ワードを含む各行の書き込みは、必ずしも順番通りに行う必要はありません。テーブル書き込みアドレスの下位 7 ビットによって、どのバッファを書き込むかが決まります。ただし各プログラミング サイクルでは、64 命令ワードの全てを書き込んで、既存データを上書きする必要があります。

テーブル書き込みプログラムメモリ アドレスによって、プログラミングする行のアドレスを選択します。プログラムメモリは、64 命令ワード間隔の「偶数」アドレス境界位置でプログラミングする必要があります。実質的に、テーブル書き込み動作の下位 7 ビットによって書き込みバッファが選択され、その行内の命令ワードがフラッシュメモリへプログラミングされます。この下位 7 ビットは 1 行をプログラミングする際には効果を持ちません。

Note: 例 5-7 のサンプルコードは、後続のサンプルコード内で Load_Write_Latch として参照されます。

例 5-7: 書き込みバッファへの読み込み

```
; Set up a pointer to the first program memory location to be written.
       MOV     #tblpage(PROG_ADDR),W0
       MOV     W0,TBLPAG
       MOV     #tbloffset(PROG_ADDR),W0

; Perform the TBLWT instructions to write the latches
; W0 is incremented in the TBLWTH instruction to point to the
; next instruction location.
       MOV     #64,W3
       MOV     #ram_image,W1
loop:
       TBLWTL  [W1++],[W0]
       TBLWTH  [W1++],[W0++]
       DEC     W3,W3
       BRA     NZ,loop
```

5.4.2.5 1 行プログラミングのサンプルコード

このサンプルコードは、複数ホールドバッファを備えるデバイスにのみ適用可能です。

まずNVMCONレジスタをフラッシュメモリの1行プログラミング向けに設定します。次にキーシーケンスを NVMKEY レジスタへ書き込んでから、WR 制御ビット (NVMCON<15>) をセットしてプログラミング動作を開始します。このキーシーケンスは、例 5-8 に厳密に従う順番で実行する必要があります。また、実行中に割り込みが発生してはなりません。従ってシーケンスを書き込む前に割り込みを有効にしておく必要があります。

CPU が動作を再開するコード位置には、2 つの NOP 命令を挿入する必要があります。シーケンスの書き込みが終了した後に、必要に応じて割り込みを有効にします。

例 5-8: 1 行プログラミング

```
; Setup NVMCON to write 1 row of program memory
       MOV     #0x4001,W0
       MOV     W0,NVMCON
; Load the 32 program memory write latches
       CALL    Load_Write_Latch(1)
; Disable interrupts, if enabled
       PUSH    SR
       MOV     #0x00E0,W0
       IOR     SR
; Write the KEY sequence
       MOV     #0x55,W0
       MOV     W0,NVMKEY
       MOV     #0xAA,W0
       MOV     W0,NVMKEY
; Start the programming sequence
       BSET    NVMCON,#WR
; Insert two NOPs after programming
       NOP
       NOP
; Re-enable interrupts, if required
       POP     SR
```

Note: 詳細は **5.4.2.4「書き込みバッファの読み込み」**を参照してください。

5.4.2.6 フラッシュメモリの 1 ワード プログラミング

ユーザ割り当てアプリケーションは既にプログラミング先のフラッシュメモリ位置を消去済みであり、テーブル書き込み命令を用いて書き込みフェッチへ 1 命令ワード (24 ビット) を書き込む場合を想定します。テーブル書き込み動作の下位 7 ビットによって書き込みラッチが選択され、その行内の 1 命令ワードがプログラミングされます。

TBLPAG レジスタにはフラッシュ アドレスの上位 8 ビットが読み込まれます。プログラミング動作中にテーブル書き込みを実行して 1 ワードをプログラミングする際に、フラッシュ アドレスの下位 16 ビットが内部で自動的に取得されます。

NVMCON レジスタをフラッシュメモリの 1 ワード プログラミング向けに設定します。次にキーシーケンスを NVMKEY レジスタへ書き込んでから、WR 制御ビット (NVMCON<15>) をセットしてプログラミング動作を開始します。このキーシーケンスは例 5-9 に厳密に従う順番で実行する必要があります。また、実行中に割り込みが発生してはなりません。従ってシーケンスを書き込む前に割り込みを無効にしておく必要があります。

CPU が動作を再開する位置には、2 つの NOP 命令を挿入する必要があります。シーケンスの書き込みが終了した後に、必要に応じて割り込みを有効にします。

例 5-9: フラッシュメモリの 1 ワード プログラミング

```
; Setup a pointer to data Program Memory
       MOV     #tblpage(PROG_ADDR),W0
       MOV     W0,TBLPAG
       MOV     #tbloffset(PROG_ADDR),W0
; Perform the TBLWT instructions to write the latches
       MOV     #LOW_WORD_N,W2
       MOV     #HIGH_BYTE_N,W3
       TBLWTL  W2,[W0]
       TBLWTH  W3,[W0++]
; Setup NVMCON for programming one word to data Program Memory
       MOV     #0x4003,W0
       MOV     W0,NVMCON
; Disable interrupts while the KEY sequence is written
       PUSH    SR
       MOV     #0x00E0,W0
       IOR     SR
; Write the key sequence
       MOV     #0x55,W0
       MOV     W0,NVMKEY
       MOV     #0xAA,W0
       MOV     W0,NVMKEY
; Start the write cycle
       BSET    NVMCON,#WR
;Re-enable interrupts, if needed
       POP     SR
```

5.4.3 デバイス設定レジスタへの書き込み

ランタイム自己プログラミング (RTSP) は、デバイス設定レジスタへの書き込みにも使用できます。RTSP を使用すると、消去サイクルを実行せずに、個々の設定レジスタの内容を書き換える事ができます。設定レジスタはシステム クロックソース、PLL 逓倍率、WDT イネーブル等、極めて重要なデバイス動作パラメータを制御します。このため、設定レジスタを書き換える際には注意が必要です。

デバイス設定レジスタのプログラミング手順は、TBLWTL 命令を必要とする点を除けば、フラッシュ プログラムメモリのプログラミング手順と同じです。これは、デバイス設定レジスタでは上位 8 ビットを使用しないためです。また、設定レジスタへアクセスするには、テーブル書き

込みアドレスのビット 23 をセットする必要があります。デバイス設定レジスタの詳細は、dsPIC33F/PIC24H ファミリ リファレンス マニュアルの**セクション 25.「デバイス設定」** (DS70194) と、各デバイス データシート内の**「特殊機能」**を参照してください。

Note 1: 一部のデバイスでは、デバイス設定レジスタへの書き込みを行えません。NVMOP<3:0> ビット定義に従って使用できるモードについては、各デバイスのデータシートを参照してください。

2: デバイス設定レジスタに対して RTSP を実行する場合は、PLL を使用せずに内部 FRC オシレータを用いてデバイスを動作させる必要があります。これ以外のクロックソースを使用してデバイスを動作させている場合は、デバイス設定レジスタに対して RTSP を実行する前に、クロックを内部 FRC オシレータへ切り換える必要があります (NOSC<2:0> = 000)。

3: オシレータ設定レジスタ (FOSC) 内のプライマリ オシレータモード選択ビット (POSCMD<1:0>) を変更する場合は、RTSP 動作を実行する前に、FOSC レジスタ内のクロック スイッチング モードビット (FCKSM<1:0>) が初期設定値 (00) になっている事を確認してください。

5.4.3.1 設定レジスタの書き込みアルゴリズム

1. TBLWTL 命令を使用して、テーブル書き込みラッチへ新たな設定値を書き込みます。
2. NVMCON レジスタを設定レジスタ書き込み向けに設定します (NVMCON = 0x4000)。
3. 割り込みが有効になっている場合は無効にします。
4. キーシーケンスを NVMKEY へ書き込みます。
5. WR ビット (NVMCON<15>) をセットして書き込みシーケンスを開始します。
6. 書き込みが終了すると CPU の動作が再開します。
7. 必要に応じて割り込みを有効にします。

例 5-10 のサンプルコードは、デバイス設定レジスタの変更に使用できます。

例 5-10:設定レジスタの書き込み

```
; Set up a pointer to the location to be written.
       MOV     #tblpage(CONFIG_ADDR),W0
       MOV     W0,TBLPAG
       MOV     #tbloffset(CONFIG_ADDR),W0
; Get the new data to write to the configuration register
       MOV     #ConfigValue,W1
; Perform the table write to load the write latch
       TBLWTL  W1,[W0]
; Configure NVMCON for a configuration register write
       MOV     #0x4000,W0
       MOV     W0,NVMCON
; Disable interrupts, if enabled
       PUSH    SR
       MOV     #0x00E0,W0
       IOR     SR
; Write the KEY sequence
       MOV     #0x55,W0
       MOV     W0,NVMKEY
       MOV     #0xAA,W0
       MOV     W0,NVMKEY
; Start the programming sequence
       BSET    NVMCON,#WR
; Insert two NOPs after programming
       NOP
       NOP
; Re-enable interrupts, if required
       POP     SR
```

5.5 レジスタマップ

フラッシュ プログラミングに関連するレジスタの概要を表 5-1 に記載します。

表 5-1: フラッシュ プログラミング レジスタ

レジスタ名	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	全リセット
NVMCON	WR	WREN	WRERR	—	—	—	—	—	—	ERASE	—	—	NVMOP<3:0>				0000
NVMKEY	—	—	—	—	—	—	—	—	NVMKEY<7:0>								0000

凡例： x = リセット時に不明の値、— = 未実装、「0」として読み出し、リセット値は 16 進表記です。

5.6 設計のヒント

質問 : **デバイスでプログラミングまたは消去を正常に行えません。コードに問題は見つかりません。どのような原因が考えられますか。**

回答 : プログラミングまたは消去サイクルを開始する前に割り込みを無効にして、キーシーケンス実行中の割り込み発生を防いでください。CPU 優先度をレベル 7 へ上げる事によって割り込みを防げます。

本書のサンプルコードでは、SR レジスタの値をスタックへ保存してから、SR レジスタと値 0x00E0 間で OR 論理演算を行って、IPL<2:0> を強制的に 111 に設定しています。優先度 7 の割り込みが発生しない場合は、DISI 命令を使用して、キーシーケンス実行中に一時的に割り込みを無効にする事もできます。

5.7 関連アプリケーション ノート

ここでは、本セクションに関連するアプリケーション ノートを紹介します。一部のアプリケーション ノートは dsPIC33F/PIC24H 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、変更が必要であったり制限事項が存在するものの利用が可能です。本モジュールに関連する最新のアプリケーション ノートは以下の通りです。

タイトル	アプリケーション ノート番号

関連するアプリケーション ノートはありません。

Note: dsPIC33F/PIC24H 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 コードについては、マイクロチップ社のウェブサイト (www.microchip.com) をご覧ください。

5.8 改訂履歴

リビジョン A (2007 年 2 月)

本書の初版

リビジョン B (2007 年 2 月)

本書全体の小規模な更新

リビジョン C (2009 年 12 月)

このリビジョンでは、dsPIC33F および PIC24H ファミリ リファレンス マニュアルの「フラッシュ プログラミング」セクションを 1 つの文書にまとめ、下の変更を加えました。

- 補足文書に関する情報を記載した網掛け注釈ボックスを本セクションの冒頭に追加
- **5.4.3「デバイス設定レジスタへの書き込み」**の Note 1 と Note 2 を追加
- 表現および体裁の変更等、本書全体の細部を修正

リビジョン D (2010 年 4 月)

このリビジョンでの更新内容

- **5.2.1.3「テーブル書き込みホールドバッファ」**に網掛け注釈ボックスを追加
- **5.2.1.4「ワードモードの書き込み」**と **5.2.1.5「バイトモードの書き込み」**の最終段落と **5.3「制御レジスタ」**の第 2 段落に、複数ホールドバッファを備えていないデバイスに関する文章を追加
- フラッシュメモリ制御レジスタ (レジスタ 5-1) に Note 3 を追加
- **5.4「ランタイム自己プログラミング (RTSP)」**の第 1 段落を更新
- **5.4.1「RTSP 動作」**の第 2 および第 3 段落を更新
- **5.4.2.1「フラッシュ行プログラミング アルゴリズム」**の第 1 段落を更新
- **5.4.2.2「フラッシュ単一ワード プログラミング アルゴリズム」**を新たに追加
- **5.4.2.4「書き込みバッファの読み込み」**に網掛け注釈ボックスを追加
- **5.4.2.5「1 行プログラミングのサンプルコード」**の第 1 段落に 1 文章を追加
- **5.4.3「デバイス設定レジスタへの書き込み」**内の網掛け注釈ボックスへ Note 1 を新たに追加
- **5.5「レジスタマップ」**を新たに追加

マイクロチップ社製デバイスのコード保護機能に関して次の点にご注意ください。

- マイクロチップ社製品は、該当するマイクロチップ社データシートに記載の仕様を満たしています。
- マイクロチップ社では、通常の条件ならびに仕様に従って使用した場合、マイクロチップ社製品のセキュリティ レベルは、現在市場に流通している同種製品の中でも最も高度であると考えています。
- しかし、コード保護機能を解除するための不正かつ違法な方法が存在する事もまた事実です。弊社の理解ではこうした手法は、マイクロチップ社データシートにある動作仕様書以外の方法でマイクロチップ社製品を使用する事になります。このような行為は知的所有権の侵害に該当する可能が非常に高いと言えます。
- マイクロチップ社は、コードの保全性に懸念を抱くお客様と連携し、対応策に取り組んでいきます。
- マイクロチップ社を含む全ての半導体メーカーで、自社のコードのセキュリティを完全に保証できる企業はありません。コード保護機能とは、マクロチップ社が製品を「解読不能」として保証するものではありません。

コード保護機能は常に進歩しています。マイクロチップ社では、常に製品のコード保護機能の改善に取り組んでいます。マクロチップ社のコード保護機能の侵害は、デジタル ミレニアム著作権法に違反します。そのような行為によってソフトウェアまたはその他の著作物に不正なアクセスを受けた場合は、デジタル ミレニアム著作権法の定めるところにより損害賠償訴訟を起こす権利があります。

商標

本書は再生紙を使用しています。

ISBN: 978-1-60932-500-8

QUALITY MANAGEMENT SYSTEM
CERTIFIED BY DNV
ISO/TS 16949:2002

マイクロチップ社では、Chandler および Tempe (アリゾナ州)、Gresham (オレゴン州) の本部、設計部およびウェハー製造工場そしてカリフォルニア州とインドのデザインセンターが ISO/TS-16949:2002 認証を取得しています。マイクロチップ社の品質システム プロセスおよび手順は、PIC®MCU および dsPIC®DSC、KEELOQ® コード ホッピング デバイス、シリアル EEPROM、マイクロペリフェラル、不揮発性メモリ、アナログ製品に採用されています。さらに、開発システムの設計と製造に関するマイクロチップ社の品質システムは ISO 9001:2000 認証を取得しています。

各国の営業所とサービス

北米

本社
2355 West Chandler Blvd.
Chandler, AZ 85224-6199
Tel: 480-792-7200
Fax:480-792-7277
技術サポート :
http://support.microchip.com
URL:
www.microchip.com

アトランタ
Duluth, GA
Tel: 678-957-9614
Fax:678-957-1455

ボストン
Westborough, MA
Tel: 774-760-0087
Fax:774-760-0088

シカゴ
Itasca, IL
Tel: 630-285-0071
Fax:630-285-0075

クリーブランド
Independence, OH
Tel: 216-447-0464
Fax:216-447-0643

ダラス
Addison, TX
Tel: 972-818-7423
Fax:972-818-2924

デトロイト
Farmington Hills, MI
Tel: 248-538-2250
Fax:248-538-2260

ココモ
Kokomo, IN
Tel: 765-864-8360
Fax:765-864-8387

ロサンゼルス
Mission Viejo, CA
Tel: 949-462-9523
Fax:949-462-9608

サンタクララ
Santa Clara, CA
Tel: 408-961-6444
Fax:408-961-6445

トロント
Mississauga, Ontario,
Canada
Tel: 905-673-0699
Fax:905-673-6509

アジア / 太平洋

アジア太平洋支社
Suites 3707-14, 37th Floor
Tower 6, The Gateway
Harbour City, Kowloon
Hong Kong
Tel: 852-2401-1200
Fax:852-2401-3431

オーストラリア - シドニー
Tel: 61-2-9868-6733
Fax:61-2-9868-6755

中国 - 北京
Tel: 86-10-8528-2100
Fax:86-10-8528-2104

中国 - 成都
Tel: 86-28-8665-5511
Fax:86-28-8665-7889

中国 - 重慶
Tel: 86-23-8980-9588
Fax:86-23-8980-9500

中国 - 香港 SAR
Tel: 852-2401-1200
Fax:852-2401-3431

中国 - 南京
Tel: 86-25-8473-2460
Fax:86-25-8473-2470

中国 - 青島
Tel: 86-532-8502-7355
Fax:86-532-8502-7205

中国 - 上海
Tel: 86-21-5407-5533
Fax:86-21-5407-5066

中国 - 瀋陽
Tel: 86-24-2334-2829
Fax:86-24-2334-2393

中国 - 深圳
Tel: 86-755-8203-2660
Fax:86-755-8203-1760

中国 - 武漢
Tel: 86-27-5980-5300
Fax:86-27-5980-5118

中国 - 西安
Tel: 86-29-8833-7252
Fax:86-29-8833-7256

中国 - 厦門
Tel: 86-592-2388138
Fax:86-592-2388130

中国 - 珠海
Tel: 86-756-3210040
Fax:86-756-3210049

アジア / 太平洋

インド - バンガロール
Tel: 91-80-3090-4444
Fax:91-80-3090-4123

インド - ニューデリー
Tel: 91-11-4160-8631
Fax:91-11-4160-8632

インド - プネ
Tel: 91-20-2566-1512
Fax:91-20-2566-1513

日本 - 横浜
Tel: 81-45-471- 6166
Fax:81-45-471-6122

韓国 - 大邱
Tel: 82-53-744-4301
Fax:82-53-744-4302

韓国 - ソウル
Tel: 82-2-554-7200
Fax:82-2-558-5932???
82-2-558-5934

マレーシア - クアラルンプール
Tel: 60-3-6201-9857
Fax:60-3-6201-9859

マレーシア - ペナン
Tel: 60-4-227-8870
Fax:60-4-227-4068

フィリピン - マニラ
Tel: 63-2-634-9065
Fax:63-2-634-9069

シンガポール
Tel: 65-6334-8870
Fax:65-6334-8850

台湾 - 新竹
Tel: 886-3-6578-300
Fax:886-3-6578-370

台湾 - 高雄
Tel: 886-7-536-4818
Fax:886-7-536-4803

台湾 - 台北
Tel: 886-2-2500-6610
Fax:886-2-2508-0102

タイ - バンコク
Tel: 66-2-694-1351
Fax:66-2-694-1350

ヨーロッパ

オーストリア - ヴェルス
Tel: 43-7242-2244-39
Fax:43-7242-2244-393

デンマーク - コペンハーゲン
Tel: 45-4450-2828
Fax:45-4485-2829

フランス - パリ
Tel: 33-1-69-53-63-20
Fax:33-1-69-30-90-79

ドイツ - ミュンヘン
Tel: 49-89-627-144-0
Fax:49-89-627-144-44

イタリア - ミラノ
Tel: 39-0331-742611
Fax:39-0331-466781

オランダ - ドリューネン
Tel: 31-416-690399
Fax:31-416-690340

スペイン - マドリッド
Tel: 34-91-708-08-90
Fax:34-91-708-08-91

イギリス - ウォーキンガム
Tel: 44-118-921-5869
Fax:44-118-921-5820

01/05/10