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 22. ダイレクト メモリアクセス (DMA)

ハイライト

本セクションには下記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 データシートの補足を目的としています。本書の内容は dsPIC33E/PIC24E 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**「ダイレクトメモリ アクセス (DMA)」**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 (http://www.microchip.com) でご覧になれます。

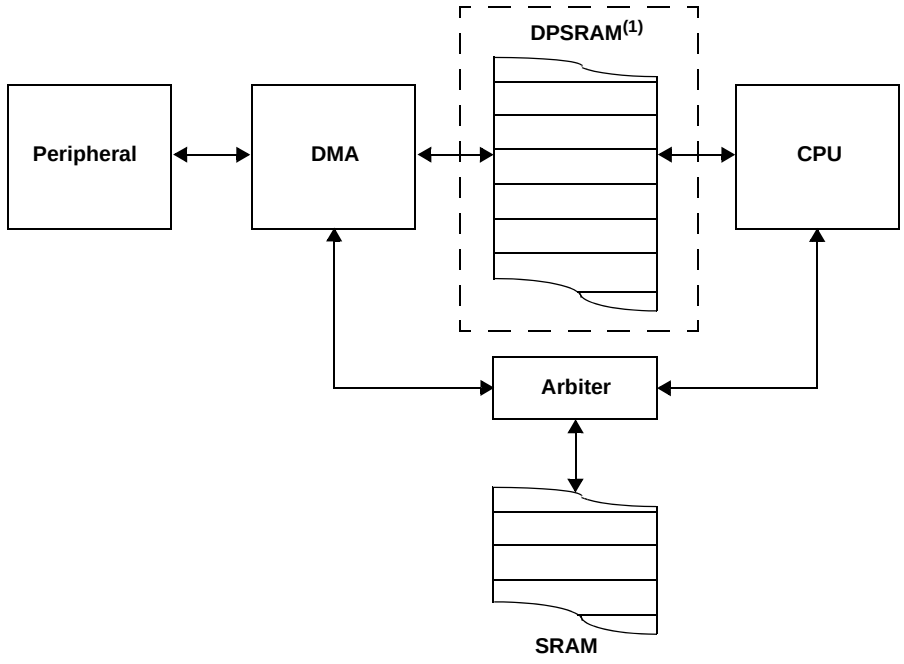
22.1 はじめに

ダイレクト メモリアクセス (DMA) コントローラは、マイクロチップ社の高性能 16 ビット デジタル シグナル コントローラ (DSC) ファミリにおける重要なサブシステムです。このサブシステムにより、CPU 時間を消費せずにメモリと周辺モジュール間でデータを転送できます。dsPIC33E/PIC24E の DMA コントローラは、決定論的機能とシステム レイテンシが重視される高性能なリアルタイム組み込みアプリケーション向けに最適化されています。

DMA コントローラは、周辺モジュールのデータレジスタとデータ空間 SRAM 間でデータを転送します。デュアルポート SRAM (DPSRAM) を備えた dsPIC33E/PIC24E デバイスの DMA サブシステムは DPSRAM とレジスタ構造を使うため、DMA は独自のアドレスとデータバスを使って CUPU 動作に影響を与える事なく動作できます。このアーキテクチャは、サイクルスチール (より優先度の高い DMA 転送が要求された際に CPU を停止させる機構) を不要にします。CPU と DMA コントローラは共に CPU ストール等の干渉を受けずにデータ空間内のアドレスに対して読み書きできるため、最大限のリアルタイム性能が得られます。一方、DMA の動作とメモリおよび周辺モジュール間のデータ転送は CPU 処理の影響を受けません。例えば、実行時自己プログラミング (RTSP) が実行中である場合、CPU は RTSP が終了するまで一切の命令を実行しません。しかし、この場合も、メモリと周辺モジュール間のデータ転送は影響を受けません。

また、DMA はデータメモリ空間の全体 (SRAM と DPSRAM) にアクセスできます。CPU または DMA のいずれかがデュアルポートではない SRAM にアクセスする場合、データメモリバスアービターを経由するため、DMA または CPU ストールが発生する可能性があります。

図 22-1: DMA コントローラ



Note 1: 一部のデバイスは DPSRAM を備えていません。ご使用になるデバイスが DPSRAM を備えているかどうかは、デバイス データシート内の**「ダイレクトメモリ アクセス (DMA)」**を参照してください。

© 2012 Microchip Technology Inc.

DMA コントローラは最大で 15 の独立したチャンネルをサポートします。各チャンネルは、一部の周辺モジュールとの単方向データ転送用に設定できます。DMA コントローラがサポートする周辺モジュールには下記が含まれます。

- ECAN™ モジュール
- データコンバータ インターフェイス (DCI)
- A/D コンバータ (ADC)
- シリアル ペリフェラル インターフェイス (SPI)
- UART (Universal Asynchronous Receiver Transmitter)
- 入力キャプチャ
- 出力コンペア
- パラレル マスタポート (PMP)

また、タイマまたは外部割り込みを使って DMA 転送を開始できます。各 DMA チャンネルは単方向です。周辺モジュールとの間で双方向 (読み出しと書き込み) の転送を行う場合、2 つの DMA チャンネルを割り当てる必要があります。複数のチャンネルがデータ転送要求を受け取った場合、チャンネル番号順の固定的な優先度に従って、転送を実行するチャンネルと保留されるチャンネルが決まります。各 DMA チャンネルは、データのブロックを転送した後に CPU に対して割り込みを生成し、そのデータブロックが処理可能である事を知らせます。

DMA コントローラは下記の機能を備えます。

- 最大 15 DMA チャンネル
- ポストインクリメント アドレッシング モードを使うレジスタ間接
- ポストインクリメント アドレッシング モードを使わないレジスタ間接
- 周辺モジュール間接アドレッシング モード (周辺モジュールが転送先アドレスを生成)
- ハーフブロックまたはフルブロック転送完了時の CPU 割り込み
- バイト転送またはワード転送
- 固定優先度のチャンネル調停
- 手動 (ソフトウェア) または自動 (周辺モジュール DMA 要求) による転送開始
- ワンショットまたは自動再送ブロック転送モード
- ピンポンモード (2 つの DPSRAM または SRAM 開始アドレスをブロック転送ごとに交互に自動切り換え)
- 各チャンネルの DMA 要求はサポートされる任意の割り込み要因から選択可能
- デバッグサポート機能

22.2 DMA レジスタ

各 DMA チャンネルには、それぞれ 6 個のステータスおよび制御レジスタが割り当てられています。

- **DMAxCON: DMA チャンネル x 制御レジスタ**

  このレジスタでは、DMA チャンネル x の有効化 / 無効化、データ転送サイズ / データ転送方向 / ブロック割り込み方法の指定、DMA チャンネル アドレッシング モード / 動作モード /NULL データ書き込みモードの選択を行います。

- **DMAxREQ: DMA チャンネル x IRQ 選択レジスタ**

  このレジスタでは、DMA チャンネル x に周辺モジュール IRQ を割り当てる事によって、その DMA チャンネルを特定の DMA 対応周辺モジュールに関連付けます。

- **DMAxSTAH: DMA チャンネル x 開始アドレスレジスタ A (HIGH)** および **DMAxSTAL: DMA チャンネル x 開始アドレスレジスタ A (LOW)**

  このレジスタでは、DMA チャンネル x と DPSRAM (または RAM) 間で転送するデータブロックのプライマリ開始アドレスを指定します。このレジスタを読み出すと、直前の転送のアドレス値が返されます。チャンネル x が有効 (アクティブ) な時にこのレジスタが書き込まれると、予測不可能な挙動が生じる可能性があるため、書き込みを回避する必要があります。

  DMAxSTA は、2 個の 16 ビットレジスタ (DMAxSTAH と DMAxSTAL) で構成された 24 ビットレジスタです。DMAxSTAH は bit 23-16、DMAxSTAL は bit 15-0 を格納します。DMAxSTA レジスタは、DMAxSTAH および DMAxSTAL レジスタを読み書きする事によってのみアクセスできます。本書では、レジスタ名として DMAxSTA と DMAxSTAH/DMAxSTAL を区別せずに使います。

- **DMAxSTBH: DMA チャンネル x 開始アドレスレジスタ B (HIGH)** および **DMAxSTBL: DMA チャンネル x 開始アドレスレジスタ B (LOW)**

  このレジスタでは、DMA チャンネル x と DPSRAM (または RAM) 間で転送するデータブロックのセカンダリ開始アドレスを指定します。このレジスタを読み出すと、直前の転送のアドレス値が返されます。チャンネル x が有効 (アクティブ) な時にこのレジスタが書き込まれると、予測不能な挙動が生じる可能性があるため、書き込みを回避する必要があります。

  DMAxSTB は、2 個の 16 ビットレジスタ (DMAxSTBH と DMAxSTBL) で構成された 24 ビットレジスタです。DMAxSTBH は bit 23-16、DMAxSTBL は bit 15-0 を格納します。DMAxSTB レジスタは、DMAxSTBH および DMAxSTBL レジスタを読み書きする事によってのみアクセスできます。本書では、レジスタ名として DMAxSTB と DMAxSTBH/DMAxSTBL を区別せずに使います。

- **DMAxPAD: DMA チャンネル x 周辺モジュール アドレス レジスタ**

  このレジスタは、周辺モジュール データレジスタの静的アドレスを格納します。対応する DMA チャンネルが有効 (アクティブ) な時にこのレジスタが書き込まれると、予測不可能な挙動が生じる可能性があるため、書き込みを回避する必要があります。

- **DMAxCNT: DMA チャンネル x 転送カウントレジスタ**

  このレジスタは転送数を格納します。チャンネルが DMAxCNT + 1 個の DMA 要求を処理すると、データブロック転送が完了したとみなされます。DMAxCNT の値が「0」の場合、1 個のデータ要素が転送されます。DMAxCNT レジスタの値は、転送データサイズ (DMAxCON レジスタの SIZE ビット) とは無関係です。対応する DMA チャンネルが有効 (アクティブ) な時にこのレジスタが書き込まれると、予測不可能な挙動が生じる可能性があるため、書き込みを回避する必要があります。

© 2012 Microchip Technology Inc.

DMA チャンネル別のレジスタ以外に、DMA コントローラには下記の 5 個の DMA ステータス レジスタが割り当てられています。

- **DSADR: 直前 DMA 転送 DPSRAM(または SRAM) アドレスレジスタ**

  全ての DMA チャンネルは、この 16 ビット読み出し専用ステータス レジスタを共有します。このレジスタは、直前の DPSRAM 読み出しまたは書き込みアクセスのアドレスを格納します。このレジスタはリセット時にクリアされます。従って、リセット後に一度も DMA 転送を実行せずにこのレジスタを読み出すと「0x0000」が返されます。このレジスタにはいつでもアクセスできますが、デバッグの支援がこのレジスタの本来の目的です。

- **DMAPWC: DMA 周辺モジュール書き込みコリジョン ステータス レジスタ (1)**

  全ての DMA チャンネルは、この 16 ビット読み出し専用ステータス レジスタを共有します。このレジスタは、周辺モジュール書き込みコリジョンフラグ PWCOLx を格納します。詳細は **22.10「データ書き込みコリジョンと要求コリジョン」**を参照してください。

- **DMARQC: DMA 要求コリジョン ステータス レジスタ (1)**

  全ての DMA チャンネルは、この 16 ビット読み出し専用ステータス レジスタを共有します。このレジスタは DMA 要求コリジョンフラグ RQCOLx を格納します。詳細は **22.10「データ書き込みコリジョンと要求コリジョン」**を参照してください。

- **DMALCA: DMA 直前アクティブ チャンネル ステータス レジスタ (1)**

  この 16 ビット読み出し専用ステータス レジスタは、直前にアクティブであった DMA チャンネルを示します。

- **DMAPPS: DMA ピンポン ステータス レジスタ (1)**

  この 16 ビット読み出し専用ステータス レジスタは、各 DMA チャンネルのピンポンモードのステータス (どちらの開始アドレスレジスタ (DMAxSTA または DMAxSTB) が選択されているか) を示します。

レジスタ 22-1: DMAxCON: DMA チャンネル x 制御レジスタ

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	U-0	U-0	U-0
CHEN	SIZE	DIR	HALF	NULLW	—	—	—
bit 15							bit 8

U-0	U-0	R/W-0	R/W-0	U-0	U-0	R/W-0	R/W-0
—	—	AMODE<1:0>		—	—	MODE<1:0>	
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 = ビットはセット	0 = ビットはクリア	x = ビットは未知

bit 15 **CHEN:** チャンネル イネーブルビット
1 = このチャンネルを有効にする
0 = このチャンネルを無効にする

bit 14 **SIZE:** データ転送サイズビット
1 = バイト
0 = ワード

bit 13 **DIR:** 転送方向ビット (ソース / デスティネーション バス選択)
1 = DPSRAM (または RAM) アドレスから読み出して周辺モジュール アドレスに書き込む
0 = 周辺モジュールアドレスから読み出して DPSRAM (または RAM) アドレスに書き込む

bit 12 **HALF:** ブロック転送割り込み選択ビット
1 = ハーフブロック (データの半分) を移動した時点で割り込みを生成する
0 = フルブロック (全てのデータ) を移動した時点で割り込みを生成する

bit 11 **NULLW:** NULL データ周辺モジュール書き込みモード選択ビット
1 = DPSRAM (または RAM) への書き込み時に周辺モジュールに NULL データを書き込む (DIR ビットがクリアされている事が必要)
0 = 通常動作

bit 10-6 **未実装 :**「0」として読み出し

bit 5-4 **AMODE<1:0>:** DMA チャンネル アドレッシング モード選択ビット
11 = 予約済み
10 = 周辺モジュール間接アドレッシング モード
01 = ポストインクリメント アドレッシング モードを使わないレジスタ間接
00 = ポストインクリメント アドレッシング モードを使うレジスタ間接

bit 3-2 **未実装 :**「0」として読み出し

bit 1-0 **MODE<1:0>:** DMA チャンネル動作モード選択ビット
11 = ワンショット (ピンポンモード有効) (各 DMA バッファとの間で 1 ブロックを転送)
10 = 連続 (ピンポンモード有効)
01 = ワンショット (ピンポンモード無効)
00 = 連続 (ピンポンモード無効)

© 2012 Microchip Technology Inc.

レジスタ 22-2: DMAxREQ: DMA チャンネル x IRQ 選択レジスタ

R/W-0	U-0	U-0	U-0	U-0	U-0	U-0	U-0
FORCE(1)	—	—	—	—	—	—	—
bit 15							bit 8

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
IRQSEL<7:0>							
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 = ビットはセット	0 = ビットはクリア	x = ビットは未知

bit 15 **FORCE:** DMA 手動転送ビット (1)
1 = 1 回の DMA 転送を実行する (手動モード)
0 = DMA 要求で自動的に DMA 転送を開始する

bit 14-8 **未実装 :**「0」として読み出し

bit 7-0 **IRQSEL<7:0>:** DMA 周辺モジュール IRQ 番号選択ビット
これらのビットは、DMA チャンネルに周辺モジュール IRQ を割り当てる事によって、その DMA チャンネルを特定の DMA 対応周辺モジュールに関連付けます。ご使用になるデバイスで利用可能なオプションについては、デバイス データ シート内の**「ダイレクトメモリ アクセス (DMA)」**を参照してください。全ての dsPIC33E/PIC24E デバイスに共通の周辺モジュールには IC、OC、ADC、SPI、ECAN、UART、DCI、PMP、INT0、タイマがあります。

Note 1: ユーザは FORCE ビットをクリアできません。FORCE ビットは、手動で開始した DMA 転送が完了するか、そのチャンネルが無効 (CHEN = 0) になった時に、ハードウェアによってクリアされます。

レジスタ 22-3: DMAxSTAH: DMA チャンネル x 開始アドレスレジスタ A (HIGH)

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

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
STA<23:16>							
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 = ビットはセット	0 = ビットはクリア	x = ビットは未知

bit 15-8 **未実装 :**「0」として読み出し

bit 7-0 **STA<23:16>:** プライマリ開始アドレスビット (ソースまたはデスティネーション)

レジスタ 22-4: DMAxSTAL: DMA チャンネル x 開始アドレスレジスタ A (LOW)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
STA<15:8>							
bit 15							bit 8

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
STA<7:0>							
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 = ビットはセット	0 = ビットはクリア	x = ビットは未知

bit 15-0 **STA<15:0>:** プライマリ開始アドレスビット (ソースまたはデスティネーション)

© 2012 Microchip Technology Inc.

レジスタ 22-5: DMAxSTBH: DMA チャンネル x 開始アドレスレジスタ B (HIGH)

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

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
STB<23:16>							
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 = ビットはセット	0 = ビットはクリア	x = ビットは未知

bit 15-8 **未実装 :** 「0」として読み出し

bit 7-0 **STB<23:16>:** プライマリ開始アドレスビット (ソースまたはデスティネーション)

レジスタ 22-6: DMAxSTBL: DMA チャンネル x 開始アドレスレジスタ B (LOW)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
STB<15:8>							
bit 15							bit 8

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
STB<7:0>							
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 = ビットはセット	0 = ビットはクリア	x = ビットは未知

bit 15-0 **STB<15:0>:** セカンダリ開始アドレスビット (ソースまたはデスティネーション)

レジスタ 22-7: DMAxPAD: DMA チャンネル x 周辺モジュール アドレス レジスタ

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
PAD<15:8>							
bit 15							bit 8

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
PAD<7:0>							
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 = ビットはセット	0 = ビットはクリア	x = ビットは未知

bit 15-0 **PAD<15:0>:** 周辺モジュール アドレスレジスタ ビット

レジスタ 22-8: DMAxCNT: DMA チャンネル x 転送カウントレジスタ

U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	CNT<13:8>					
bit 15							bit 8

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
CNT<7:0>							
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 = ビットはセット	0 = ビットはクリア	x = ビットは未知

bit 15-14 **未実装 :**「0」として読み出し

bit 13-0 **CNT<13:0>:** DMA 転送カウントレジスタ ビット

レジスタ 22-9: DSADR: 直前 DMA 転送 DPSRAM(または SRAM) アドレスレジスタ

R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0
DSADR<15:8>							
bit 15							bit 8

R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0
DSADR<7:0>							
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 = ビットはセット	0 = ビットはクリア	x = ビットは未知

bit 15-0 **DSADR<15:0>:** DMA がアクセスした直前の DMA アドレスビット

© 2012 Microchip Technology Inc.

レジスタ 22-10: DMAPWC: DMA 周辺モジュール書き込みコリジョン ステータス レジスタ (1)

U-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0
—	PWCOL14	PWCOL13	PWCOL12	PWCOL11	PWCOL10	PWCOL9	PWCOL8
bit 15							bit 8

R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0
PWCOL7	PWCOL6	PWCOL5	PWCOL4	PWCOL3	PWCOL2	PWCOL1	PWCOL0
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 = ビットはセット	0 = ビットはクリア	x = ビットは未知

bit 15 **未実装 :** 「0」として読み出し

bit 14 **PWCOL14:** チャンネル 14 周辺モジュール書き込みコリジョン フラグビット
1 = 書き込みコリジョンを検出した
0 = 書き込みコリジョンは検出していない

bit 13 **PWCOL13:** チャンネル 13 周辺モジュール書き込みコリジョン フラグビット
1 = 書き込みコリジョンを検出した
0 = 書き込みコリジョンは検出していない

bit 12 **PWCOL12:** チャンネル 12 周辺モジュール書き込みコリジョン フラグビット
1 = 書き込みコリジョンを検出した
0 = 書き込みコリジョンは検出していない

bit 11 **PWCOL11:** チャンネル 11 周辺モジュール書き込みコリジョン フラグビット
1 = 書き込みコリジョンを検出した
0 = 書き込みコリジョンは検出していない

bit 10 **PWCOL10:** チャンネル 10 周辺モジュール書き込みコリジョン フラグビット
1 = 書き込みコリジョンを検出した
0 = 書き込みコリジョンは検出していない

bit 9 **PWCOL9:** チャンネル 9 周辺モジュール書き込みコリジョン フラグビット
1 = 書き込みコリジョンを検出した
0 = 書き込みコリジョンは検出していない

bit 8 **PWCOL8:** チャンネル 8 周辺モジュール書き込みコリジョン フラグビット
1 = 書き込みコリジョンを検出した
0 = 書き込みコリジョンは検出していない

bit 7 **PWCOL7:** チャンネル 7 周辺モジュール書き込みコリジョン フラグビット
1 = 書き込みコリジョンを検出した
0 = 書き込みコリジョンは検出していない

bit 6 **PWCOL6:** チャンネル 6 周辺モジュール書き込みコリジョン フラグビット
1 = 書き込みコリジョンを検出した
0 = 書き込みコリジョンは検出していない

bit 5 **PWCOL5:** チャンネル 5 周辺モジュール書き込みコリジョン フラグビット
1 = 書き込みコリジョンを検出した
0 = 書き込みコリジョンは検出していない

bit 4 **PWCOL4:** チャンネル 4 周辺モジュール書き込みコリジョン フラグビット
1 = 書き込みコリジョンを検出した
0 = 書き込みコリジョンは検出していない

bit 3 **PWCOL3:** チャンネル 3 周辺モジュール書き込みコリジョン フラグビット
1 = 書き込みコリジョンを検出した
0 = 書き込みコリジョンは検出していない

Note 1: DMA チャンネルの数は製品によって異なります。各ビットは 1 つの DMA チャンネルに対応します。ご使用になるデバイスの DMA チャンネル数については、デバイス データシート内の**「ダイレクトメモリアクセス (DMA)」**を参照してください。

レジスタ 22-10: DMAPWC: DMA 周辺モジュール書き込みコリジョン ステータス レジスタ (1) (続き)

bit 2 **PWCOL2:** チャンネル 2 周辺モジュール書き込みコリジョン フラグビット
1 = 書き込みコリジョンを検出した
0 = 書き込みコリジョンは検出していない

bit 1 **PWCOL1:** チャンネル 1 周辺モジュール書き込みコリジョン フラグビット
1 = 書き込みコリジョンを検出した
0 = 書き込みコリジョンは検出していない

bit 0 **PWCOL0:** チャンネル 0 周辺モジュール書き込みコリジョン フラグビット
1 = 書き込みコリジョンを検出した
0 = 書き込みコリジョンは検出していない

Note 1: DMA チャンネルの数は製品によって異なります。各ビットは 1 つの DMA チャンネルに対応します。ご使用になるデバイスの DMA チャンネル数については、デバイス データシート内の**「ダイレクトメモリアクセス (DMA)」**を参照してください。

© 2012 Microchip Technology Inc.

レジスタ 22-11: DMARQC: DMA 要求コリジョン ステータス レジスタ (1)

U-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0
—	RQCOL14	RQCOL13	RQCOL12	RQCOL11	RQCOL10	RQCOL9	RQCOL8
bit 15							bit 8

R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0
RQCOL7	RQCOL6	RQCOL5	RQCOL4	RQCOL3	RQCOL2	RQCOL1	RQCOL0
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 = ビットはセット	0 = ビットはクリア	x = ビットは未知

bit 15 **未実装 :** 「0」として読み出し

bit 14 **RQCOL14:** チャンネル 14 転送要求コリジョン フラグビット
1 = FORCE ビットによる手動転送要求と割り込みによる転送要求のコリジョンを検出した
0 = 要求コリジョンは検出していない

bit 13 **RQCOL13:** チャンネル 13 転送要求コリジョン フラグビット
1 = FORCE ビットによる手動転送要求と割り込みによる転送要求のコリジョンを検出した
0 = 要求コリジョンは検出していない

bit 12 **RQCOL12:** チャンネル 12 転送要求コリジョン フラグビット
1 = FORCE ビットによる手動転送要求と割り込みによる転送要求のコリジョンを検出した
0 = 要求コリジョンは検出していない

bit 11 **RQCOL11:** チャンネル 11 転送要求コリジョン フラグビット
1 = FORCE ビットによる手動転送要求と割り込みによる転送要求のコリジョンを検出した
0 = 要求コリジョンは検出していない

bit 10 **RQCOL10:** チャンネル 10 転送要求コリジョン フラグビット
1 = FORCE ビットによる手動転送要求と割り込みによる転送要求のコリジョンを検出した
0 = 要求コリジョンは検出していない

bit 9 **RQCOL9:** チャンネル 9 転送要求コリジョン フラグビット
1 = FORCE ビットによる手動転送要求と割り込みによる転送要求のコリジョンを検出した
0 = 要求コリジョンは検出していない

bit 8 **RQCOL8:** チャンネル 8 転送要求コリジョン フラグビット
1 = FORCE ビットによる手動転送要求と割り込みによる転送要求のコリジョンを検出した
0 = 要求コリジョンは検出していない

bit 7 **RQCOL7:** チャンネル 7 転送要求コリジョン フラグビット
1 = FORCE ビットによる手動転送要求と割り込みによる転送要求のコリジョンを検出した
0 = 要求コリジョンは検出していない

bit 6 **RQCOL6:** チャンネル 6 転送要求コリジョン フラグビット
1 = FORCE ビットによる手動転送要求と割り込みによる転送要求のコリジョンを検出した
0 = 要求コリジョンは検出していない

bit 5 **RQCOL5:** チャンネル 5 転送要求コリジョン フラグビット
1 = FORCE ビットによる手動転送要求と割り込みによる転送要求のコリジョンを検出した
0 = 要求コリジョンは検出していない

bit 4 **RQCOL4:** チャンネル 4 転送要求コリジョン フラグビット
1 = FORCE ビットによる手動転送要求と割り込みによる転送要求のコリジョンを検出した
0 = 要求コリジョンは検出していない

Note 1: DMA チャンネルの数は製品によって異なります。各ビットは 1 つの DMA チャンネルに対応します。ご使用になるデバイスの DMA チャンネル数については、デバイス データシート内の**「ダイレクトメモリアクセス (DMA)」**を参照してください。

レジスタ 22-11: DMARQC: DMA 要求コリジョン ステータス レジスタ (1) (続き)

bit 3 **RQCOL3:** チャンネル 3 転送要求コリジョン フラグビット
1 = FORCE ビットによる手動転送要求と割り込みによる転送要求のコリジョンを検出した
0 = 要求コリジョンは検出していない

bit 2 **RQCOL2:** チャンネル 2 転送要求コリジョン フラグビット
1 = FORCE ビットによる手動転送要求と割り込みによる転送要求のコリジョンを検出した
0 = 要求コリジョンは検出していない

bit 1 **RQCOL1:** チャンネル 1 転送要求コリジョン フラグビット
1 = FORCE ビットによる手動転送要求と割り込みによる転送要求のコリジョンを検出した
0 = 要求コリジョンは検出していない

bit 0 **RQCOL0:** チャンネル 0 転送要求コリジョン フラグビット
1 = FORCE ビットによる手動転送要求と割り込みによる転送要求のコリジョンを検出した
0 = 要求コリジョンは検出していない

Note 1: DMA チャンネルの数は製品によって異なります。各ビットは 1 つの DMA チャンネルに対応します。ご使用になるデバイスの DMA チャンネル数については、デバイス データシート内の**「ダイレクトメモリアクセス (DMA)」**を参照してください。

© 2012 Microchip Technology Inc.

レジスタ 22-12: DMALCA: DMA 直前アクティブ チャンネル ステータス レジスタ (1)

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	R-1	R-1	R-1	R-1
—	—	—	—	LSTCH<3:0>			
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 = ビットはセット	0 = ビットはクリア	x = ビットは未知

bit 15-4 **未実装 :** 「0」として読み出し

bit 3-0 **LSTCH<3:0>:** 直前アクティブ DMA チャンネル ステータスビット
1111 = システムリセット後に DMA 転送は発生していない
1110 = 直前のデータ転送はチャンネル 14 で発生した
1101 = 直前のデータ転送はチャンネル 13 で発生した
1100 = 直前のデータ転送はチャンネル 12 で発生した
1011 = 直前のデータ転送はチャンネル 11 で発生した
1010 = 直前のデータ転送はチャンネル 10 で発生した
1001 = 直前のデータ転送はチャンネル 9 で発生した
1000 = 直前のデータ転送はチャンネル 8 で発生した
0111 = 直前のデータ転送はチャンネル 7 で発生した
0110 = 直前のデータ転送はチャンネル 6 で発生した
0101 = 直前のデータ転送はチャンネル 5 で発生した
0100 = 直前のデータ転送はチャンネル 4 で発生した
0011 = 直前のデータ転送はチャンネル 3 で発生した
0010 = 直前のデータ転送はチャンネル 2 で発生した
0001 = 直前のデータ転送はチャンネル 1 で発生した
0000 = 直前のデータ転送はチャンネル 0 で発生した

Note 1: DMA チャンネルの数は製品によって異なります。ご使用になるデバイスの DMA チャンネル数については、デバイス データシート内の**「ダイレクトメモリ アクセス (DMA)」**を参照してください。

レジスタ 22-13: DMAPPS: DMA ピンポン ステータス レジスタ (1)

U-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0
—	PPST14	PPST13	PPST12	PPST11	PPST10	PPST9	PPST8
bit 15							bit 8

R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0
PPST7	PPST6	PPST5	PPST4	PPST3	PPST2	PPST1	PPST0
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 = ビットはセット	0 = ビットはクリア	x = ビットは未知

bit 15 **未実装 :**「0」として読み出し

bit 14 **PPST14:** チャンネル 14 ピンポンモード ステータスフラグ ビット
1 = DMASTB14 レジスタが選択されている
0 = DMASTA14 レジスタが選択されている

bit 13 **PPST13:** チャンネル 13 ピンポンモード ステータスフラグ ビット
1 = DMASTB13 レジスタが選択されている
0 = DMASTA13 レジスタが選択されている

bit 12 **PPST12:** チャンネル 12 ピンポンモード ステータスフラグ ビット
1 = DMASTB12 レジスタが選択されている
0 = DMASTA12 レジスタが選択されている

bit 11 **PPST11:** チャンネル 11 ピンポンモード ステータスフラグ ビット
1 = DMASTB11 レジスタが選択されている
0 = DMASTA11 レジスタが選択されている

bit 10 **PPST10:** チャンネル 10 ピンポンモード ステータスフラグ ビット
1 = DMASTB10 レジスタが選択されている
0 = DMASTA10 レジスタが選択されている

bit 9 **PPST9:** チャンネル 9 ピンポンモード ステータスフラグ ビット
1 = DMASTB9 レジスタが選択されている
0 = DMASTA9 レジスタが選択されている

bit 8 **PPST8:** チャンネル 8 ピンポンモード ステータスフラグ ビット
1 = DMASTB8 レジスタが選択されている
0 = DMASTA8 レジスタが選択されている

bit 7 **PPST7:** チャンネル 7 ピンポンモード ステータスフラグ ビット
1 = DMASTB7 レジスタが選択されている
0 = DMASTA7 レジスタが選択されている

bit 6 **PPST6:** チャンネル 6 ピンポンモード ステータスフラグ ビット
1 = DMASTB6 レジスタが選択されている
0 = DMASTA6 レジスタが選択されている

bit 5 **PPST5:** チャンネル 5 ピンポンモード ステータスフラグ ビット
1 = DMASTB5 レジスタが選択されている
0 = DMASTA5 レジスタが選択されている

bit 4 **PPST4:** チャンネル 4 ピンポンモード ステータスフラグ ビット
1 = DMASTB4 レジスタが選択されている
0 = DMASTA4 レジスタが選択されている

Note 1: DMA チャンネルの数は製品によって異なります。各ビットは 1 つの DMA チャンネルに対応します。ご使用になるデバイスの DMA チャンネル数については、デバイス データシート内の**「ダイレクトメモリ アクセス (DMA)」**を参照してください。

© 2012 Microchip Technology Inc.

レジスタ 22-13: DMAPPS: DMA ピンポン ステータス レジスタ (1) (続き)

bit 3 **PPST3:** チャンネル 3 ピンポンモード ステータスフラグ ビット
1 = DMASTB3 レジスタが選択されている
0 = DMASTA3 レジスタが選択されている

bit 2 **PPST2:** チャンネル 2 ピンポンモード ステータスフラグ ビット
1 = DMASTB2 レジスタが選択されている
0 = DMASTA2 レジスタが選択されている

bit 1 **PPST1:** チャンネル 1 ピンポンモード ステータスフラグ ビット
1 = DMASTB1 レジスタが選択されている
0 = DMASTA1 レジスタが選択されている

bit 0 **PPST0:** チャンネル 0 ピンポンモード ステータスフラグ ビット
1 = DMASTB0 レジスタが選択されている
0 = DMASTA0 レジスタが選択されている

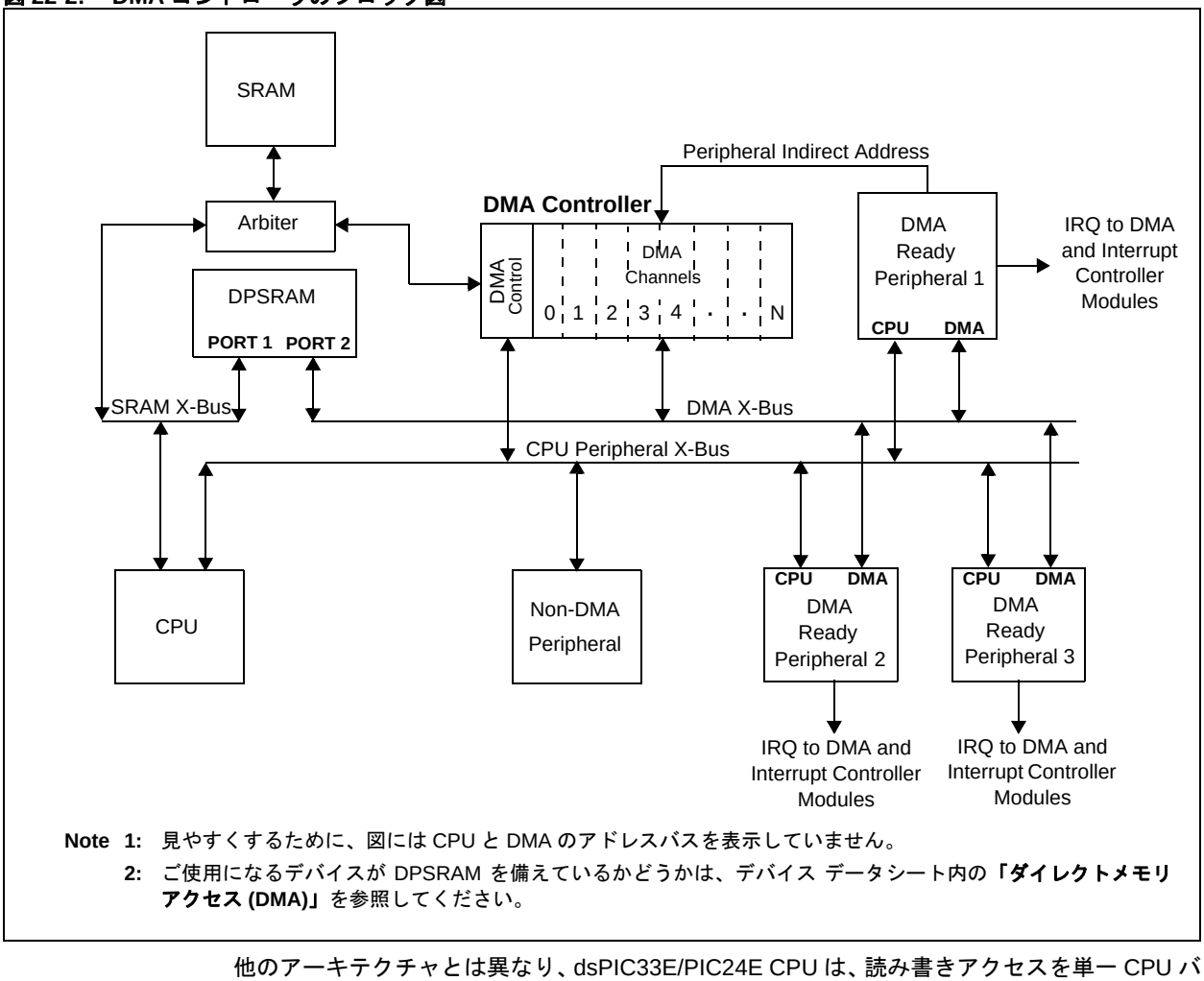
Note 1: DMA チャンネルの数は製品によって異なります。各ビットは 1 つの DMA チャンネルに対応します。ご使用になるデバイスの DMA チャンネル数については、デバイス データシート内の**「ダイレクトメモリアクセス (DMA)」**を参照してください。

22.3 DMA のブロック図

図 22-2 のブロック図は、dsPIC33E/PIC24E の内部アーキテクチャに DMA がどのように組み込まれているのかを示しています。CPU は、デュアルポート SRAM (DPSRAM) ブロックのポート 1 と通信するために、通常の SRAM との通信に使用するのと同じ X バスを使います。周辺モジュールとの通信には、これとは別に X データ空間内に存在する周辺モジュール用 X バスを使います。

デュアルポート SRAM を備えたデバイスの場合、各 DMA チャンネルは、DPSRAM のポート 2 および各DMA対応周辺モジュールのDMAポートと通信するために、専用のDMAバスを使います。

図 22-2: DMA コントローラのブロック図 (1,2)



Note 1: 見やすくするために、図には CPU と DMA のアドレスバスを表示していません。

2: ご使用になるデバイスが DPSRAM を備えているかどうかは、デバイス データシート内の**「ダイレクトメモリアクセス (DMA)」**を参照してください。

他のアーキテクチャとは異なり、dsPIC33E/PIC24E CPU は、読み書きアクセスを単一 CPU バスサイクル内で完了できます。同様に DMA も、専用バスを使って単一バスサイクル内でバイトまたはワードの転送を完了できます。従って、全ての DMA 転送は実行中に割り込まれません。あるチャンネルで始まった転送は、他チャンネルの動作に関係なく、そのサイクル中に完了します。

また、DMA はデータメモリ空間の全体 (SRAM と DPSRAM) にアクセスできます。CPU または DMA のいずれかがデュアルポートではない SRAM にアクセスする場合、データメモリバスアービターを経由するため、DMA または CPU ストールが発生する可能性があります。

ユーザ アプリケーションは、任意の DMA 対応周辺モジュール割り込みを DMA 要求として指定できます。DMA 要求とは、DMA に対する IRQ の事です。ある DMA チャンネルを特定の割り込み (DMA 要求) に応答するように設定した場合、対応する CPU 割り込みを無効にする必要があります。これを無効にしないと、CPU 割り込みも要求されます。

ソフトウェアから手動で各 DMA チャンネルをトリガする事もできます。DMAxCON レジスタの FORCE ビットをセットすると、手動 DMA 要求が生成されます。この DMA 要求は、割り込みによる DMA 要求と同様に調停されます (**22.8「DMA チャンネルの調停とオーバーラン」**)。

© 2012 Microchip Technology Inc.

22.4 DMA データ転送

図 22-3 に、周辺モジュールとデュアルポート SRAM 間のテータ転送を示します。

A. この例では、DMA チャンネル 5 を DMA 対応周辺モジュール 1 に割り当てています。

B. 周辺モジュールは、データ転送の準備が完了すると DMA 要求を発行します。DMA 要求は、同時に存在する他の DMA 要求との間で調停されます。調停時にこのチャンネルが最高優先度である場合、次のサイクル中に転送が完了します。その時点では優先度が他より低い場合、この DMA 要求は最高優先度になるまで保留されます。

C. DMA チャンネルは、ユーザ アプリケーションがそのチャンネルに対して定義した周辺モジュール アドレスからデータを読み出します。

D. DMA チャンネルは、読み出したデータを指定された DPSRAM アドレスに書き込みます。

この図は、レジスタ間接モードでの動作を示しています。この場合、DPSRAM アドレスは、DMA レジスタ (DMAxSTA または DMAxSTB) によって、DMA チャンネル側で指定されます。周辺モジュール間接モードの場合、DPSRAM アドレスは DMA チャンネル側ではなく周辺モジュール側から指定されます。詳細は **22.6.6「周辺モジュール間接アドレッシング モード」**を参照してください。

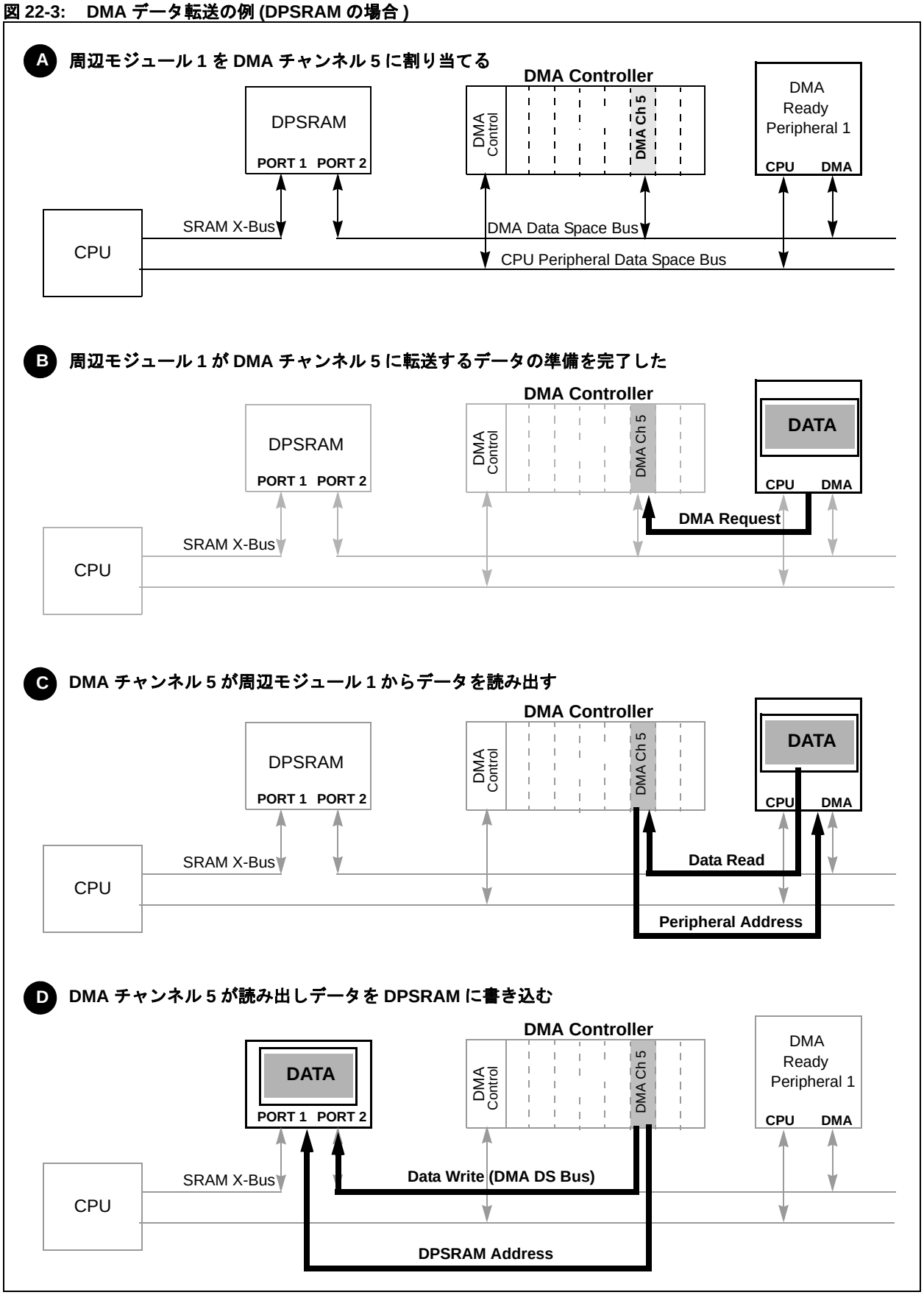
DMA 読み書き転送動作は、単一命令サイクル内で割り込まれる事なく完了します。この処理中、データ転送が完了するまで DMA 要求はその DMA チャンネルでラッチされます。

この間、DMA チャンネルは転送カウンタレジスタ (DMA5CNT) を監視し、転送カウンタ値がユーザ アプリケーションによって定義されたリミット値に達すると、データ転送が完了したとみなして CPU 割り込みを生成し、受信したデータを処理するよう CPU に促します。

スループットを最大限に高めるために、DMA コントローラはデータ転送サイクル中に保留中または後続の DMA 要求の調停を続けます。

周辺モジュールと SRAM 間のデータ転送は基本的に図 22-3 に示した例と同じですが、SRAM への全てのアクセスはアービターを経由するため、DMA または CPU ストールが発生する可能性があるという点で異なります。

図 22-3: DMA データ転送の例 (DPSRAM の場合)



© 2012 Microchip Technology Inc.

22.5 DMA のセットアップ

DMA データ転送を正しく実行するために、DMA チャンネルと周辺モジュールを下記に従って適切に設定する必要があります。

- DMA チャンネルを周辺モジュールに割り当てる (**22.5.1「周辺モジュールへの DMA チャンネルの割り当て」**参照)
- 周辺モジュールを正しく設定する (**22.5.2「周辺モジュールのコンフィグレーション」**参照)
- DPSRAM (または RAM) のデータ開始アドレスを初期化する (**22.5.3「メモリアドレスの初期化」**参照)
- DMA 転送カウンタを初期化する (**22.5.4「DMA 転送カウントのセットアップ」**参照)
- アドレッシング モードと動作モードを適切に選択する (**22.6「DMA の動作モード」**参照)

22.5.1 周辺モジュールへの DMA チャンネルの割り当て

DMA チャンネルには、読み出し元または書き込み先の周辺モジュール アドレスと、転送要求源 (周辺モジュール IRQ 番号) および転送要求方法 (手動 / 自動) を設定する必要があります。これらは、それぞれ DMA チャンネル x 周辺モジュール アドレスレジスタ (DMAxPAD) と DMA チャンネル x IRQ 選択レジスタ (DMAxREQ) で設定します。

DMA チャンネルを特定の周辺モジュールに割り当てるには、これらのレジスタに表 22-1 に示す値を書き込む必要があります。

表 22-1: 周辺モジュールへの DMA チャンネルの割り当て (1)

DMA に割り当てる周辺モジュール	DMAxREQ レジスタ IRQSEL<7:0> ビット	周辺モジュールから読み出す場合の DMAxPAD レジスタ値	周辺モジュールに書き込む場合の DMAxPAD レジスタ値
INT0 – 外部割り込み 0	00000000	—	—
IC1 – 入力キャプチャ 1	00000001	0x0144 (IC1BUF)	—
IC2 – 入力キャプチャ 2	00000101	0x014C (IC2BUF)	—
IC3 – 入力キャプチャ 3	00100101	0x0154 (IC3BUF)	—
IC4 – 入力キャプチャ 4	00100110	0x015C (IC4BUF)	—
OC1 – 出力コンペア 1	00000010	—	0x0906 (OC1R) 0x0904 (OC1RS)
OC2 – 出力コンペア 2	00000110	—	0x0910 (OC2R) 0x090E (OC2RS)
OC3 – 出力コンペア 3	00011001	—	0x091A (OC3R) 0x0918 (OC3RS)
OC4 – 出力コンペア 4	00011010	—	0x0924 (OC4R) 0x0922 (OC4RS)
TMR2 – Timer2	00000111	—	—
TMR3 – Timer3	00001000	—	—
TMR4 – Timer4	00011011	—	—
TMR5 – Timer5	00011100	—	—
SPI1 転送完了	00001010	0x0248 (SPI1BUF)	0x0248 (SPI1BUF)
SPI2 転送完了	00100001	0x0268 (SPI2BUF)	0x0268 (SPI2BUF)
SPI3 転送完了	01011011	0x02A8 (SPI3BUF)	0x02A8 (SPI3BUF)
SPI4 転送完了	01111011	0x02C8 (SPI4BUF)	0x02C8 (SPI4BUF)
UART1RX – UART1 受信	00001011	0x0226 (U1RXREG)	—
UART1TX – UART1 送信	00001100	—	0x0224 (U1TXREG)
UART2RX – UART2 受信	00011110	0x0236 (U2RXREG)	—
UART2TX – UART2 送信	00011111	—	0x0234 (U2TXREG)
UART3RX – UART3 受信	01010010	0x0256 (U3RXREG)	—
UART3TX – UART3 送信	01010011	—	0x0254 (U3TXREG)
UART4RX – UART4 受信	01011000	0x02B6 (U4RXREG)	—
UART4TX – UART4 送信	01011001	—	0x02B4 (U4TXREG)
ECAN1 – RX データレディ	00100010	0x0440 (C1RXD)	—
ECAN1 – TX データ要求	01000110	—	0x0442 (C1TXD)
ECAN2 – RX データレディ	00110111	0x0540 (C2RXD)	—
ECAN2 – TX データ要求	01000111	—	0x0542 (C2TXD)
DCI – CODEC 転送完了	00111100	0x0290 (RXBUF0)	0x0298 (TXBUF0)
ADC1 – ADC1 変換完了	00001101	0x0300 (ADC1BUF0)	—
ADC2 – ADC2 変換完了	00010101	0x0340 (ADC2BUF0)	—
PMP – PMP データ移動	00101101	0x0608 (PMDIN1)	0x0608 (PMDIN1)

Note 1: DMA 対応周辺モジュールの一覧は、デバイス データシート内の**「ダイレクトメモリ アクセス (DMA)」**に記載しています。

2 つの DMA チャンネルに対して同じ周辺モジュールを DMA 要求源として割り当てた場合、両方のチャンネルが同時に DMA 要求を受け取ります。ただし、優先度が高い方のチャンネルが先に転送を実行し、他方のチャンネルは保留されます。このような状況は、1 つの DMA 要求を使って周辺モジュール (SPI 等) との間で双方向にデータを転送する場合に発生します。この場合、2 つの DMA チャンネルを使い、1 チャンネルを周辺モジュールからの読み出し用に割り当て、もう 1 チャンネルを周辺モジュールへのデータ書き込み用に割り当てます。両チャンネルは同一の DMA 要求を使います。

DMAxPAD レジスタが表 22-1 以外の値に初期化された場合、その周辺モジュール アドレスへの DMA チャンネル書き込みは無視されます。そのアドレスからの DMA チャンネル読み出しの結果は「0」です。

© 2012 Microchip Technology Inc.

22.5.2 周辺モジュールのコンフィグレーション

DMA セットアップの第 2 段階として、DMA 対応周辺モジュールを DMA 動作用に適切に設定する必要があります。表 22-2 に、各 DMA 対応周辺モジュールの設定要件の概要を示します。

表 22-2: DMA 対応周辺モジュールの設定における留意事項

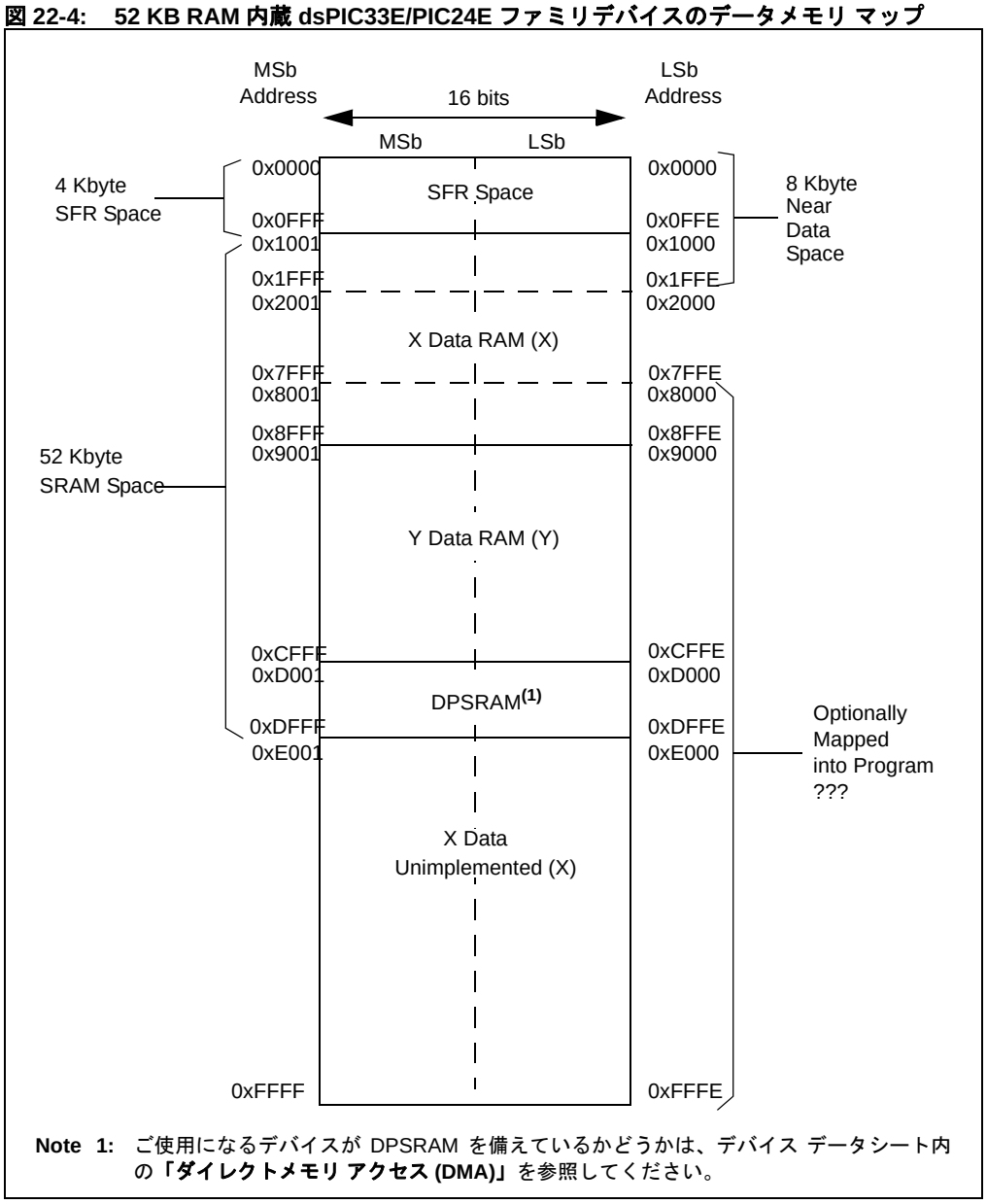
DMA 対応 周辺モジュール	設定時の留意事項
ECAN™	ECAN バッファは DMA によって管理されます。ユーザは、CAN バッファ領域と FIFO の合計サイズを、ECAN FIFO 制御レジスタ (C1FCTRL) の DMA バッファサイズ ビット (DMABS<2:0>) で定義する必要があります。例 22-9 のサンプルコードを参照してください。
データコンバータ インターフェイス (DCI)	DCI は、データワードをバッファリングするたびに割り込みを生成するよう設定する必要があります。このため、DCI 制御 2 レジスタ (DCICON2) のバッファ長制御ビット (BLEN<1:0>) を「00」に設定します。双方向のデータ転送 (RX と TX) をサポートするために、2 つの DMA チャンネルに、DMA 要求として同じ DCI 割り込みを割り当てる必要があります。DCI モジュールがマスタとして動作してデータを受信するだけの場合でも、2 つの DMA チャンネルを使って片方のチャンネルでダミーデータを送信する必要があります。例 22-11 のサンプルコードを参照してください。
10/12 ビット A/D コンバータ (ADC)	ADC を周辺モジュール間接アドレッシング モードに設定した DMA と一緒に使う場合、ADCx 制御 2 レジスタ (ADCxCON2) の DMA アドレス インクリメント レートビット (SMPI<4:0>) と DCx 制御 4 レジスタ (ADCxCON4) のアナログ入力あたり DMA バッファサイズ ビット (DMABL<2:0>) を適切に設定する必要があります。さらに、ADC アドレスを生成するために、ADCx 制御 1 レジスタ (ADxCON1) の DMA バッファ構築モードビット (ADDMABM) を適切に設定する必要があります。詳細は **22.6.6.1「ADC の DMA アドレス生成サポート」**を参照してください。例 22-5 と例 22-7 のサンプルコードを参照してください。
シリアル ペリフェラル インターフェイス (SPI)	SPI モジュールがマスタとして動作してデータを受信するだけの場合でも、2 つの DMA チャンネルを割り当てて片方のチャンネルでダミーデータを送信する必要があります。別の方法として、NULL データ書き込みモードで 1 つの DMA チャンネルだけを使う事もできます。詳細は **22.6.11「NULL データ書き込みモード」**を参照してください。例 22-12 のサンプルコードを参照してください。
UART	UART は、各キャラクタを受信または送信するたびに割り込みを生成するように設定する必要があります。UART レシーバは各キャラクタを受信するたびに RX 割り込みを生成する必要があります。このため、ステータス / 制御レジスタ (UxSTA) の受信割り込みモード選択ビット (URXISEL<1:0>) を「00」または「01」に設定する必要があります。 UART トランシーバは各キャラクタを送信するたびに TX 割り込みを生成する必要があります。このため、ステータス / 制御レジスタ (UxSTA) の送信割り込みモード選択ビット (UTXISEL0 と UTXISEL1) を「0」に設定する必要があります。例 22-10 のサンプルコードを参照してください。 ステータスビットを監視する場合、UART レシーバ用の DMA チャンネルはワードモードに設定する必要があります。詳細は dsPIC33E/PIC24E ファミリ リファレンス マニュアルの**セクション 17.「UART」**(DS70582) を参照してください。
入力キャプチャ	入力キャプチャ モジュールは、キャプチャ イベントが発生するたびに割り込みを生成する必要があります。このため、入力キャプチャ制御レジスタ (ICxCON) の割り込みあたりキャプチャ数ビット (ICI<1:0>) を「00」に設定する必要があります。例 22-4 のサンプルコードを参照してください。
出力コンペア	出力コンペア モジュールでは、DMA 転送用に特別な設定は不要です。ただし、通常は DMA 要求の生成にタイマを使うため、タイマを適正に設定する必要があります。例 22-3 のサンプルコードを参照してください。
外部割り込みとタイマ	DMA 要求として選択できるのは、外部割り込み 0 と Timer2 および Timer3 だけです。これらの周辺モジュール自体は DMA 転送をサポートしませんが、DMA 対応周辺モジュールの DMA 転送をトリガするために使えます。例えば、Timer2 は PWM モードで出力コンペア周辺モジュールの DMA トランザクションをトリガできます。例 22-3 のサンプルコードを参照してください。

DMA 対応周辺モジュールでエラー条件が発生すると、通常はステータスフラグがセットされて割り込みが生成されます (ユーザ アプリケーションでそれらの割り込みを有効にしている場合のみ)。CPU が周辺モジュールをサービスしている時、データ割り込みハンドラでエラーフラグをチェックし、必要に応じて適切に対処する必要があります。これに対し、DMA チャンネルが周辺モジュールをサービスしている時、DMA はデータ転送要求にのみ応答する事ができ、後続のエラー条件を一切認識しません。DMA 対応周辺モジュール内のエラー条件に関連する割り込みを全て有効にし、発生したそれらの割り込みはユーザ定義割り込みサービスルーチン (ISR) で処理する必要があります。

22.5.3 メモリアドレスの初期化

DMA セットアップの第 3 段階として 、DMA アクセス用のメモリバッファを DPSRAM または SRAM 空間に割り当てる必要があります。DPSRAM 領域が利用可能かどうかと、そのアドレスおよびサイズは、dsPIC33E/PIC24E ファミリのデバイスごとに異なります。詳細はデバイス データシート内の**「ダイレクトメモリ アクセス (DMA)」**を参照してください。図 22-4 に、52 Kbyte の RAM を内蔵した dsPIC33E/PIC24E デバイスにおける 4 Kbyte の DPSRAM 領域を示します。

図 22-4: 52 KB RAM 内蔵 dsPIC33E/PIC24E ファミリデバイスのデータメモリ マップ



Note 1: ご使用になるデバイスが DPSRAM を備えているかどうかは、デバイス データシート内の**「ダイレクトメモリ アクセス (DMA)」**を参照してください。

CPU は、DPSRAM (DMA RAM) および通常の RAM 領域を含むメモリ領域全体にアクセスできます。DPSRAM には専用バスを使うため、DMA モジュールはアービターによる一切の遅延を生じる事なく DPSRAM にアクセスできます。CPU または DMA のいずれかがデュアルポートではない SRAM にアクセスする場合、データメモリバス アービターを経由するため、DMA または CPU ストールが発生する可能性があります。

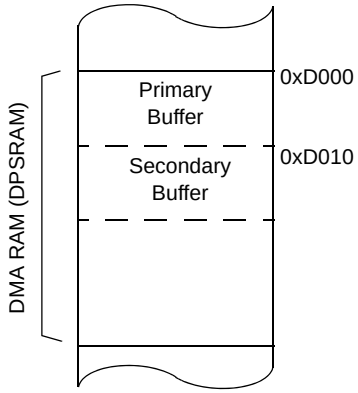
DMA モジュールを正しく動作させるために、読み書きアクセス先の DPSRAM または RAM アドレスを DMA モジュールに知らせる必要があります。この情報は、DMA チャンネル x 開始アドレスレジスタ A (DMAxSTAH と DMAxSTAL) と、DMA チャンネル x 開始アドレスレジスタ B (DMAxSTBH と DMAxSTBL) で設定します。

© 2012 Microchip Technology Inc.

図 22-5 に、例として dsPIC33E/PIC24E 上の DMA チャンネル 4 のプライマリおよびセカンダリ バッファがそれぞれアドレス 0xD000 と 0xD010 に配置されている状態を示します。

このアドレス情報をアプリケーションにハードコードするには、使用するデバイスのメモリレイアウトを熟知している必要があります。また、DMA 転送完了後にこれらのバッファへアクセスするには、ポインタ計算が必要です。従って、これらの設定を異なるデバイスに簡単に移植する事はできません。

図 22-5: DMA メモリ内のプライマリおよびセカンダリ バッファの割り当て



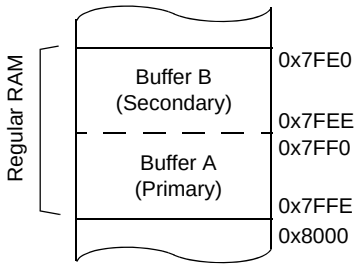
Code Example:

```
DMA4STAL = 0xD000;
DMA4STAH = 0x0000;
DMA4STBL = 0xD010;
DMA4STBH = 0x0000;
```

このため dsPIC® 向け MPLAB® C コンパイラは、DMA バッファの初期化とアクセスを単純化するために、ビルトイン C プリミティブを提供します。図 22-6 に記載したコードは、通常のデータメモリ内に 2 つのバッファを配置し、DMA チャンネルのアクセス先アドレスをそれらの位置に初期化します。図 22-7 に記載したコードは、拡張データ空間内の DMA メモリ内に 2 つのバッファを配置し、DMA チャンネルのアクセス先アドレスをそれらの位置に初期化します。

図 22-6: MPLAB® IDE によるプライマリ / セカンダリ DMA バッファの割り当て - Case 1

Case 1: バッファを通常の RAM 領域に配置する



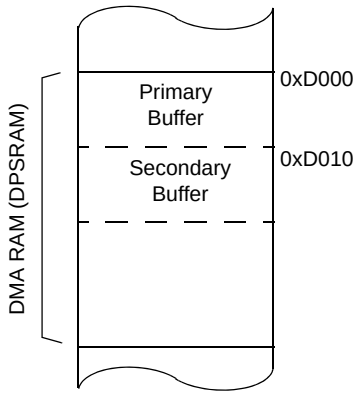
Code Example:

```
unsigned int BufferA[8];
unsigned int BufferB[8];

DMA0STAH = 0x0000;
DMA0STAL = &BufferA;
DMA0STBH = 0x0000;
DMA0STBL = &BufferB;
```

Note: バッファが拡張データ領域内にある場合、__eds__ タグを使ってバッファを宣言する必要があります。

図 22-7: MPLAB® IDE によるプライマリ / セカンダリ DMA バッファの割り当て - Case 2

Case 2: バッファを拡張データ領域内の DMA RAM に配置する



Code Example:

```
__eds__ unsigned int BufferA[8] __attribute__((eds,space(dma)));
__eds__ unsigned int BufferB[8] __attribute__((eds,space(dma)));


DMA0STAH = 0x0000;
DMA0STAL = __builtin_dmaoffset(BufferA);
DMA0STBH = 0x0000;
DMA0STBL = __builtin_dmaoffset(BufferB);
```

Note: DMA RAM が拡張データ領域内ではない場合、バッファの宣言に __eds__ タグと eds 属性は不要です。

DMAxSTA (および / または DMAxSTB) レジスタが適正な値に初期化されなかったために、DMA チャンネルがアクセス可能空間の外側の RAM アドレスに対して読み書きを行う結果となる場合、そのアドレスに対する DMA 書き込みは無視されます。そのメモリアドレスからの DMA チャンネルの読み出し結果は「0」です。DMA モジュールが未実装メモリアドレスにアクセスを試みると、DMA アドレスエラー ソフトトラップが発行され、DMA アドレスエラー ソフトトラップ ステータスビット (DAE) がセットされます。

22.5.4 DMA 転送カウントのセットアップ

DMA セットアップの第 4 段階では、データブロック転送を完了するまでに処理する必要のある DMA 要求の数 (N + 1) を各 DMA チャンネルに設定する必要があります。値「N」は、DMA チャンネル x 転送カウントレジスタ (DMAxCNT) で指定します。DMAxCNT の値が「0」の場合、1 個のデータ要素が転送されます。

DMAxCNT レジスタの値は、DMAxCON レジスタの SIZE ビットで指定するデータ転送サイズ (バイトまたはワード) とは無関係です。

DMAxCNT レジスタが適正な値に初期化されなかったために、DMA チャンネルがアクセス可能空間の外側の RAM アドレスに対して読み書きを行う結果となる場合、そのアドレスに対する DMA 書き込みは無視されます。そのメモリアドレスからの DMA チャンネルの読み出し結果は「0」です。

22.5.5 動作モードのセットアップ

最後に、DMA セットアップの第 5 段階として、DMA チャンネル x 制御レジスタ (DMAxCON) で各 DMA チャンネルの動作モードを指定します。詳細は **22.6「DMA の動作モード」**を参照してください。

© 2012 Microchip Technology Inc.

22.6 DMA の動作モード

DMA チャンネルは以下の動作モードをサポートします。

- ワードサイズまたはバイトサイズのデータ転送
- 転送方向 (周辺モジュールからメモリへ、または、メモリから周辺モジュールへ)
- CPU への割り込みタイミング (フルブロック (全データ) 転送時、またはハーフブロック (半分のデータ) 転送時)
- ポストインクリメントまたは静的なメモリ アドレッシング
- 周辺モジュール間接アドレッシング
- ワンショットまたは連続ブロック転送
- 2 つの開始アドレス オフセット (DMAxSTA と DMAxSTB) の自動的な交互切り換え (ピンポンモード)
- NULL データ書き込みモード

さらに、DMA 転送を 1 回だけ実行する手動モードも使えます。

22.6.1 ワードサイズまたはバイトサイズのデータ転送

各 DMA チャンネルは、ワードサイズまたはバイトサイズのデータ転送用に設定できます。ワードデータの転送は、偶数アドレスに対してのみ可能です。バイトデータの転送は、偶数アドレスでも奇数アドレスでも可能です。

SIZE ビット (DMAxCON<14>) がクリアすると、ワードサイズのデータを転送します。この場合、ポストインクリメント アドレッシング モードを使うレジスタ間接を有効にすると、1 ワードを転送するたびにアドレスが 2 ずつポストインクリメントします (**22.6.5「ポストインクリメント アドレッシング モードを使わないレジスタ間接」**参照)。

SIZE ビット (DMAxCON<14>) をセットすると、バイトサイズのデータを転送します。この場合、ポストインクリメント アドレッシング モードを使うレジスタ間接を有効にすると、1 バイトを転送するたびにアドレスが 1 ずつポストインクリメントします。

22.6.2 転送方向

各 DMA チャンネルには転送方向 (周辺モジュール→ DPSRAM/RAM、または、DPSRAM/RAM →周辺モジュール) を設定できます。

DMAxCON レジスタの転送方向ビット (DIR) をクリアすると、周辺モジュール (DMAxPAD が指定する周辺モジュールアドレス) から読み出したデータを DMAxSTA または DMAxSTB が指定する DPSRAM/RAM アドレスに書き込みます。

DIR ビットをセットすると、DMAxSTA または DMAxSTB が指定する DPSRAM/RAM アドレスから読み出したデータを周辺モジュール (DMAxPAD が指定する周辺モジュール アドレス) に書き込みます。

設定済みの各チャンネルは単方向データパスとして機能します。従って、周辺モジュールがデータ読み出しと書き込みの両方向に DMA モジュールを使う場合、2 チャンネル (読み出し用に 1 チャンネルと書き込み用に 1 チャンネル) を割り当てる必要があります。

22.6.3 フルブロックまたはハーフブロック転送割り込み

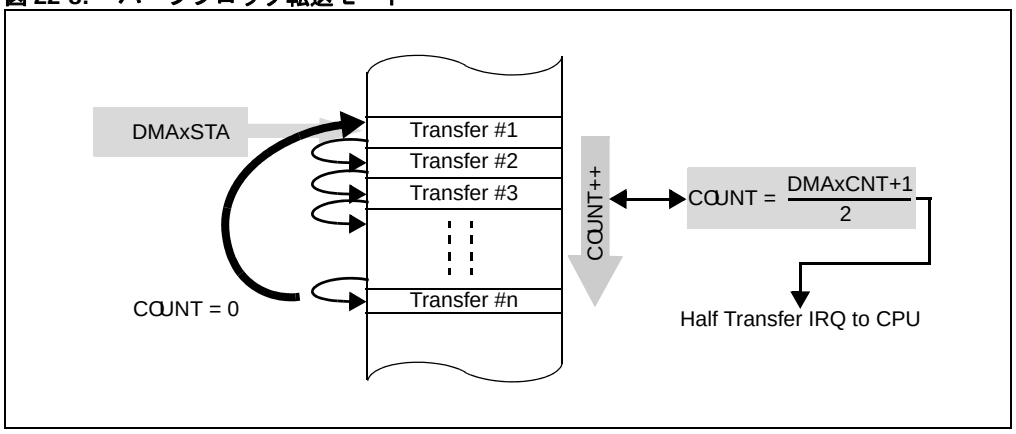
各 DMA チャンネルは、ブロックデータの全部を転送した時または半分を転送した時に割り込みコントローラに対して割り込みを生成します。どちらのタイミングで割り込むかは、DMA チャンネル x 制御レジスタ (DMAxCON) の HALF ビットを使って下記のように選択できます。

HALF = 0: フルブロック (全てのデータ) を転送した時に割り込みを生成する

HALF = 1: ハーフブロック (半分のデータ) を転送した時に割り込みを生成する

DMA 連続モードを使う場合、CPU は DMA の転送速度以上の速度で受信または送信データを処理する必要があります。ハーフブロック転送割り込みを使ってデータの半分を転送した時に割り込みを生成する事により、この要求を軽減できます。例えば、DMA コントローラが ADC を連続的に読み出す場合にハーフブロック転送割り込みを使うと、CPU はバッファが完全にフルになる前にバッファを処理できます。CPU によるバッファ処理が DMA によるバッファ書き込みを追い越さない限り、この方法によって CPU の応答時間に対する要求を軽減する事ができます。図 22-8 に、このプロセスを示します。

図 22-8:　ハーフブロック転送モード



HALF ビットをセットすると、DMA は常にバッファ A および / またはバッファ B の前半が転送された時にのみ割り込みを生成します。バッファ A および / または B の全データ転送完了時には割り込みは生成されません。つまり、DMA が (DMAxCNT + 1)/2 回のデータ転送を完了した時にのみ割り込みが生成されます。(DMAxCNT + 1) が奇数である場合、割り込みは (DMAxCNT + 2)/2 回のデータ転送後に生成されます。

例えばDMA3をワンショット/ピンポンバッファ モード(MODE<1:0> = 11)に設定し、DMA3CNT = 7 とした場合、DMA3 割り込みは 2 回 (バッファ A から 4 個のエレメント転送後と、バッファ B から 4 個のエレメント転送後に) 生成されます。詳細は **22.6.7「ワンショット モード」**と **22.6.9「ピンポンモード」**を参照してください。

DMA チャンネルはハーフブロック転送後またはフルブロック転送後のいずれか一方だけで割り込みを生成しますが、各 DMA 割り込み中にユーザ アプリケーションで HALF ビットの値をトグルする事により、ハーフブロック転送後とフルブロック転送後の両方で DMA 割り込みを生成できます。例として、DMA チャンネルの HALF ビットが「1」(ハーフブロック転送後に割り込みを生成) にセットされている場合、割り込みサービス中にユーザ アプリケーションで HALF ビットを「0」にリセットすると、その DMA チャンネルはフルブロック転送後にも割り込みを生成します。

これらの割り込みを有効にするために、割り込みコントローラ モジュールで割り込みイネーブル制御レジスタ (IECx) の対応する DMA 割り込みイネーブルビット (DMAxIE) をセットしておく必要があります。

例 22-1 に、DMA チャンネル 0 割り込みを有効にするサンプルコードを示します。

例 22-1: DMA チャンネル 0 割り込みを有効にするコード

```
IEC0bits.DMA0IE = 1;
```

DMA チャンネル転送割り込みが発生すると、割り込みフラグステータス レジスタ (IFSx) 内の対応するステータスフラグがセットされ、これにより ISR がトリガされます。転送完了 ISR の再実行を防ぐために、ユーザ アプリケーションでこのステータスフラグをクリアする必要があります。

例として、DMA チャンネル 0 割り込みを有効にしている場合、DMA チャンネル 0 の転送が完了すると、対応する割り込みが割り込みコントローラに向けて発行されます。ステータスフラグをクリアして ISR の再実行を防ぐために、下記のコードを DMA チャンネル 0 の ISR に含める必要があります。

例 22-2: DMA チャンネル 0 割り込みをクリアするコード

```
void __attribute__((__interrupt__,no_auto_psv)) _DMA0Interrupt(void)
{
  .
  .
  .
IFS0bits.DMA0IF = 0;
}
```


© 2012 Microchip Technology Inc.

22.6.4 ポストインクリメント アドレッシング モードを使うレジスタ間接

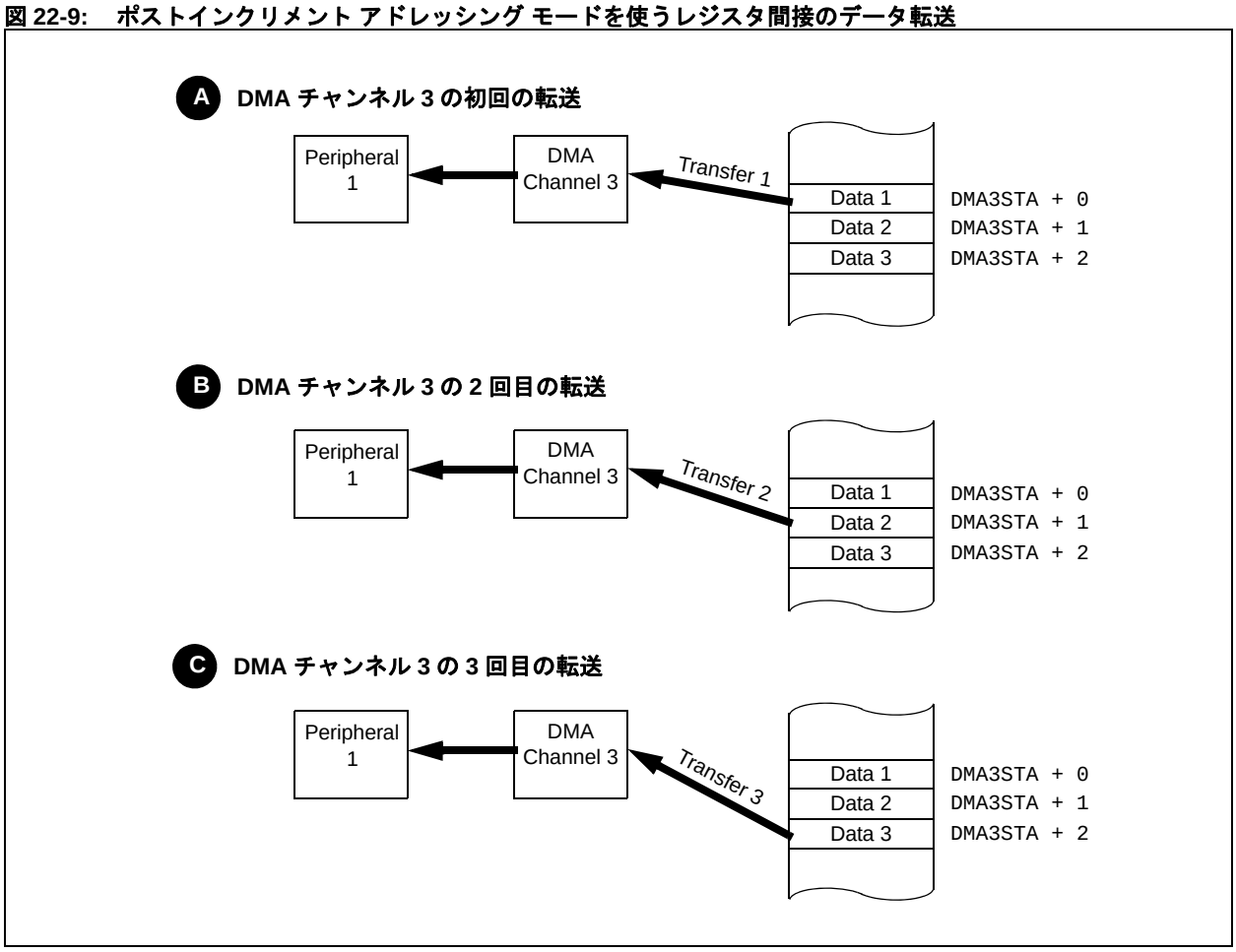
ポストインクリメント アドレッシング モードを使うレジスタ間接では、毎回の転送後に DPSRAM/RAM アドレスをインクリメントする事によってデータのブロックを移動します。

DMA コントローラのリセット時に、既定値として DMA チャンネルはこのモードに設定されます。DMA チャンネル制御レジスタ (DMAxCON) のアドレッシング モード選択ビット (AMODE<1:0>) を「00」に設定すると、このモードが選択されます。このモードでは、開始アドレスレジスタ (DMAxSTA または DMAxSTB) によってメモリバッファの開始アドレスが指定されます。

ユーザ アプリケーションは、開始アドレスレジスタを読み出す事によって、直前の転送アドレスを特定できます。DMA コントローラはこのレジスタの内容を変更しません。

図 22-9 に、このモードでのデータ転送を示します。

図 22-9: ポストインクリメント アドレッシング モードを使うレジスタ間接のデータ転送



例 22-3: ポストインクリメント アドレッシング モードを使うレジスタ間接による出力コンペア データの DMA 転送

出力コンペア 1 モジュールを PWM モードに設定する :

```
OC1CON1 = 0;              // Reset OC module
OC1CON2 = 0;
OC1R = 0x60;              // Initialize PWM Duty Cycle
OC1RS = 0x60;             // Initialize PWM Duty Cycle Buffer

OC1CONbits.OCM = 6;       // Configure OC for the PWM mode
```

DMA チャンネル 3 を、Timer2 を DMA 要求源とするポストインクリメント モードに設定する :

```
__eds__ unsigned int BufferA[32] __attribute__((eds));
/* Insert code here to initialize BufferA with desired Duty Cycle values */

DMA3CONbits.AMODE = 0;    // Configure DMA for Register Indirect mode
                          // with post-increment
DMA3CONbits.MODE = 0;     // Configure DMA for Continuous mode
DMA3CONbits.DIR   = 1;    // RAM-to-Peripheral data transfers
DMA3PAD = (volatile unsigned int)&OC1RS; // Point DMA to OC1RS
DMA3CNT = 31;             // 32 DMA request
DMA3REQ = 7;              // Select Timer2 as DMA request source

DMA3STAL = __builtin_dmaoffset(BufferA);
DMA3STAH = 0x0000;

IFS2bits.DMA3IF = 0;      // Clear the DMA Interrupt Flag bit
IEC2bits.DMA3IE = 1;      // Set the DMA Interrupt Enable bit

DMA3CONbits.CHEN = 1;     // Enable DMA
```

Timer2 を出力コンペア PWM モードに設定する :

```
PR2 = 0xBF;               // Initialize PWM period
T2CONbits.TON = 1;        // Start Timer2
```

DMA チャンネル 3 割り込みハンドラを設定する :

```
void __attribute__((__interrupt__,no_auto_psv)) _DMA3Interrupt(void)
{
 /* Update BufferA with new Duty Cycle values if desired here */

 IFS2bits.DMA3IF = 0;     //Clear the DMA3 Interrupt Flag
}
```


© 2012 Microchip Technology Inc.

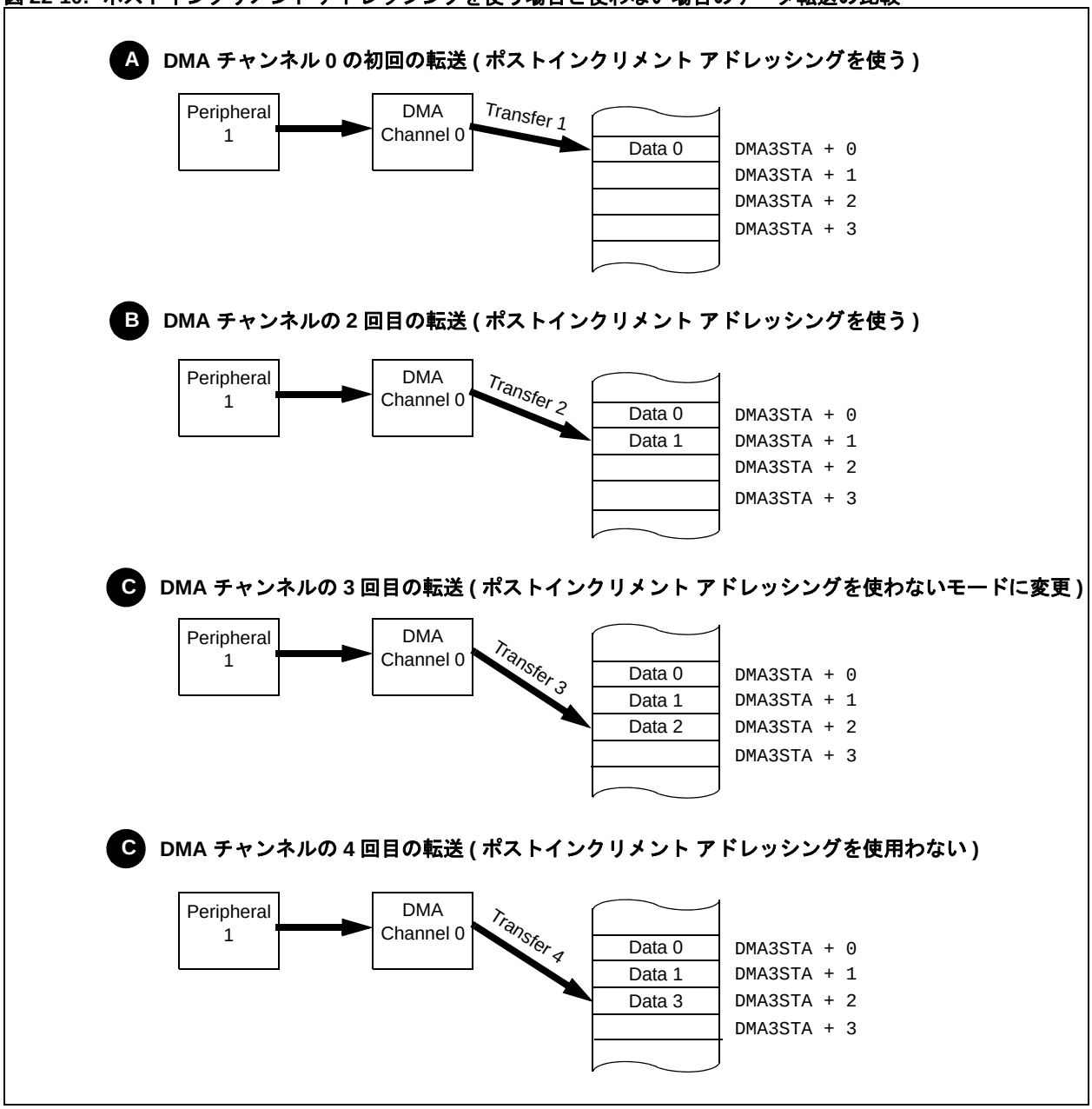
22.6.5 ポストインクリメント アドレッシング モードを使わないレジスタ間接

ポストインクリメント アドレッシング モードを使わないレジスタ間接では、毎回の転送後にデータバッファの開始アドレスをインクリメントせずにデータのブロックを移動します。このモードでは、開始アドレスレジスタ (DMAxSTA または DMAxSTB) によってメモリバッファの開始アドレスが指定されます。DMA データ転送の実行中、メモリアドレスは次のアドレスへインクリメントしません。従って、次回の DMA データ転送も同じメモリアドレスに対して行われます。

DMAチャンネル制御レジスタ(DMAxCON)のアドレッシング モード選択ビット(AMODE<1:0>)を「01」に設定すると、このモードが選択されます。

DMA チャンネルがアクティブな状態 (DMA 転送を何回か実行した後等) でアドレッシングモードをポストインクリメント アドレッシングモードを使わないレジスタ間接に変更した場合、DMA アドレスは現在のバッファ位置を指します (つまり、DMAxSTA または DMAxSTB の値は、現在のバッファ位置と異なる可能性があります)。図 22-10 に、周辺モジュールからメモリへのデータ転送で、ポストインクリメント アドレッシングを使う場合と使わない場合の動作の比較を示します。

図 22-10: ポストインクリメント アドレッシングを使う場合と使わない場合のデータ転送の比較



例 22-4: ポストインクリメント アドレッシング モードを使わないレジスタ間接による入力キャプチャ データの DMA 転送

入力キャプチャ 1 を DMA 動作用に設定する :

```
IC1CON1 = 0;           // Reset IC module
IC1CON2 = 0;
IC1CONbits.ICTMR = 1;  // Select Timer2 contents for capture
IC1CONbits.ICM   = 2;  // Capture every falling edge
IC1CONbits.ICI   = 0;  // Generate DMA request on every capture event
```

Timer2 を入力キャプチャ モジュール用に設定する :

```
PR2 = 0xBF;            // Initialize count value
T2CONbits.TON = 1;     // Start timer
```

DMA チャンネル 0 を「ポストインクリメント アドレッシングを使わない」モードに設定する :

```
unsigned int CaptureValue;

DMA0CONbits.AMODE = 1; // Configure DMA for Register indirect
                       // without post-increment
DMA0CONbits.MODE = 0;  // Configure DMA for Continuous mode
DMA0PAD = (volatile unsigned int)&IC1BUF; // Point DMA to IC1BUF
DMA0CNT = 0;           // Interrupt after each transfer
DMA0REQ = 1;           // Select Input Capture module as DMA request source

DMA3STAL = __builtin_dmaoffset(BufferA);
DMA3STAH = 0x0000;

IFS0bits.DMA0IF = 0;   // Clear the DMA Interrupt Flag bit
IEC0bits.DMA0IE = 1;   // Set the DMA Interrupt Enable bit

DMA0CONbits.CHEN = 1;  // Enable DMA
```

DMA チャンネル 0 割り込みハンドラを設定する :

```
void __attribute__((__interrupt__,no_auto_psv)) _DMA3Interrupt(void)
{
 /* Process CaptureValue variable here */

 IFS0bits.DMA0IF = 0; // Clear the DMA3 Interrupt Flag
}
```

22.6.6 周辺モジュール間接アドレッシング モード

周辺モジュール間接アドレッシング モードは、DMA チャンネル側ではなく周辺モジュール側で DPSRAM/RAM アドレスの可変部を制御する特殊なアドレッシング モードです。このモードでは、周辺モジュールが DPSRAM/RAM アドレスの下位ビット (LSb) を生成し、DMA チャンネルは固定されたバッファ ベースアドレスを提供します。ただし、このモードにおいても、DMA チャンネルは引き続きデータ転送の調整 / 転送数のカウント / 対応する CPU 割り込みの生成を行います。

周辺モジュール間接アドレッシング モードでも、周辺モジュール側の要求に応じて、どちらの方向にもデータを転送できます。従って DMA チャンネルは、周辺モジュールの読み出しまたは書き込みのいずれか用に適正に設定する必要があります。

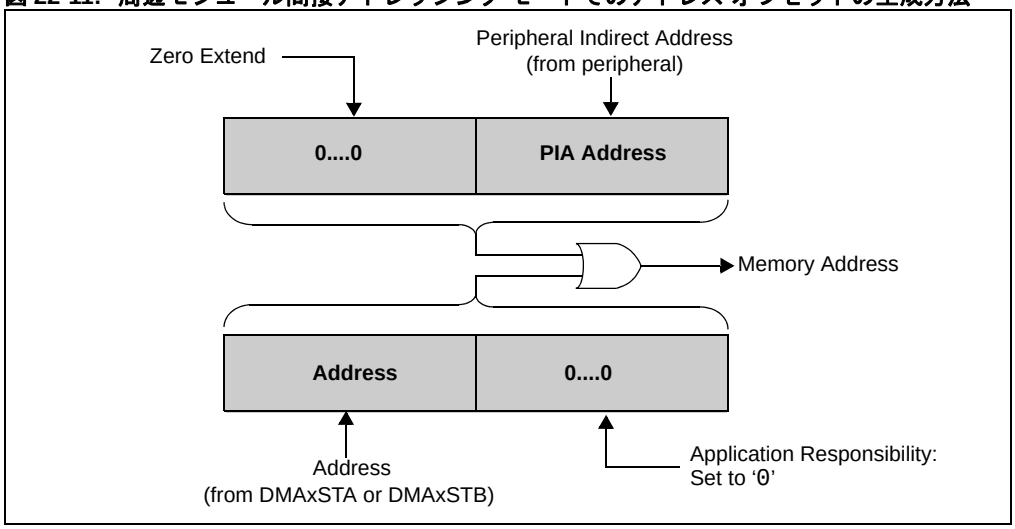
DMAチャンネル制御レジスタ(DMAxCON)のアドレッシング モード選択ビット(AMODE<1:0>)を「1x」に設定すると、周辺モジュール間接アドレッシング モードが選択されます。

周辺モジュール間接アドレッシング モードでは、このモードをサポートする周辺モジュール側の要求に合わせて DMA 動作を設定できます。すなわち、メモリ内のデータへのアクセス先アドレス シーケンスは、周辺モジュール側で定義されます。例えば ADC からの読み出しデータを複数バッファに並び換えて格納する事により、後の CPU 処理負荷を軽減できます。

周辺モジュールが周辺モジュール間接アドレッシング モードをサポートしている場合、その周辺モジュールからの DMA 要求割り込みが発生すると、周辺モジュールから DMA チャンネルにアドレスが渡されます。この要求に応答する DMA チャンネル側でも周辺モジュール間接アドレッシングが有効になっていれば、バッファのベースアドレスとゼロ拡張した周辺モジュール間接アドレスの論理和 (OR) によって実際のメモリアドレスが生成されます (図 22-11 参照)。

© 2012 Microchip Technology Inc.

図 22-11: 周辺モジュール間接アドレッシング モードでのアドレス オフセットの生成方法



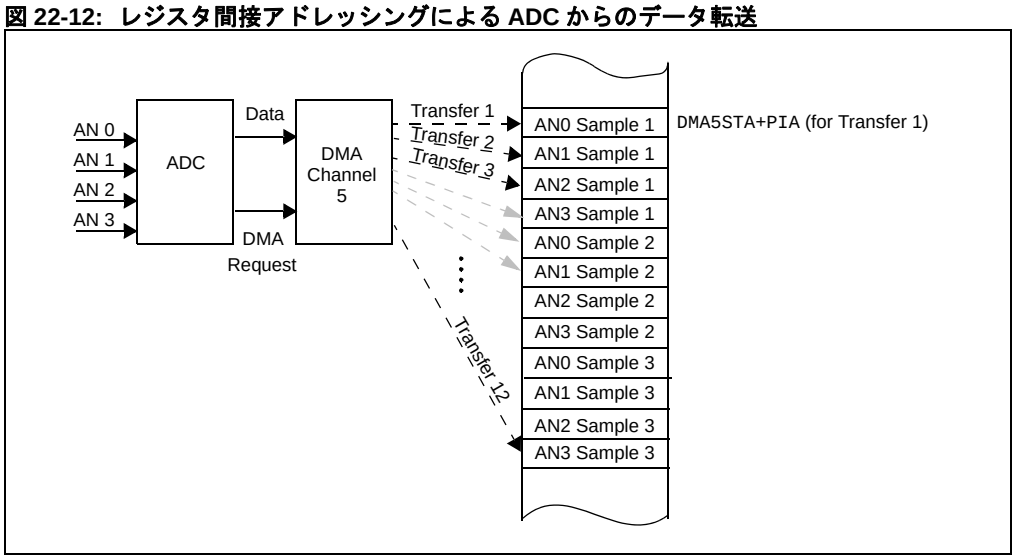
周辺モジュールが生成するアドレスの下位ビット数は、周辺モジュールによって決まります。アプリケーション プログラムは、メモリ内バッファのベースアドレスを選択する必要があります。このベースアドレスの下位ビット (周辺モジュールが生成する下位ビットに対応するビット) はゼロである事が必要です。他のモードと同様に、DMA 開始アドレスレジスタを読み出すと、直前のメモリ転送アドレスが返されますが、これには上記のアドレス計算が反映されます。DMA チャンネルが周辺モジュール間接アドレッシング用に設定されていない場合、周辺モジュールからのアドレスは無視され、データ転送は通常モードで行われます。

周辺モジュール間接アドレッシング モードは、他の全ての動作モードと互換性を持ち、現在のところ ADC および ECAN モジュールでサポートされます。

22.6.6.1 ADC の DMA アドレス生成サポート

周辺モジュール間接アドレッシング モードでは、周辺モジュールがアドレッシング シーケンスを定義します。これにより、アドレッシング シーケンスを周辺モジュールの機能に適合させる事ができます。例として、入力 0 ～ 3 を順番に繰り返し (例 : 0, 1, 2, 3, 0, 1, ...) 変換するよう設定した ADC に、ポストインクリメント アドレッシング モードを使うレジスタ間接に設定した DMA チャンネルを割り当てた場合、DMA 転送は ADC データを連続したバッファ位置に格納します (図 22-12 参照)。例 22-5 に、このコンフィグレーション用のサンプルコードを示します。

図 22-12: レジスタ間接アドレッシングによる ADC からのデータ転送



例 22-5: レジスタ間接アドレッシングによる ADC からのデータ転送

ADC1 をチャンネル 0 ～ 3 でサンプリングするように設定する :

```
AD1CON1bits.FORM = 3;     // Data Output Format:Signed Fraction (Q15 format)
AD1CON1bits.SSRC = 2;     // Sample Clock Source:GP Timer starts conversion
AD1CON1bits.ASAM = 1;     // Sampling begins immediately after conversion
AD1CON1bits.AD12B = 0;    // 10-bit ADC operation
AD1CON1bits.SIMSAM = 0;   // Samples individual channels sequentially

AD1CON2bits.BUFM = 0;
AD1CON2bits.CSCNA = 1;    // Scan CH0+ Input Selections during Sample A bit
AD1CON2bits.CHPS = 0;     // Converts CH0

AD1CON3bits.ADRC = 0;     // ADC Clock is derived from Systems Clock
AD1CON3bits.ADCS = 63;    // ADC Conversion Clock

AD1CON4bits.ADDMAEN = 1;
//AD1CHS0:Analog-to-Digital Input Select Register
AD1CHS0bits.CH0SA = 0;    // MUXA +ve input selection (AIN0) for CH0
AD1CHS0bits.CH0NA = 0;    // MUXA -ve input selection (VREF-) for CH0

//AD1CHS123:Analog-to-Digital Input Select Register
AD1CHS123bits.CH123SA = 0;// MUXA +ve input selection (AIN0) for CH1
AD1CHS123bits.CH123NA = 0;// MUXA -ve input selection (VREF-) for CH1

//AD1CSSH/AD1CSSL:Analog-to-Digital Input Scan Selection Register
AD1CSSH = 0x0000;
AD1CSSL = 0x000F;         // Scan AIN0, AIN1, AIN2, AIN3 inputs
```

Timer3 を ADC1 の変換トリガ用に設定する :

```
TMR3 = 0x0000;
PR3 = 4999;               // Trigger ADC1 every 125 ms @ 40 MIPS
IFS0bits.T3IF = 0;        // Clear Timer3 interrupt
IEC0bits.T3IE = 0;        // Disable Timer3 interrupt

T3CONbits.TON = 1;        // Start Timer3
```

DMA チャンネル 5 をポストインクリメント アドレッシング モードを使うレジスタ間接に設定する :

```
__eds__ unsigned int BufferA[32] __attribute__((eds,space(dma)));
__eds__ unsigned int BufferB[32] __attribute__((eds,space(dma)));

DMA5CONbits.AMODE = 0;    // Configure DMA for Register Indirect mode
                          // with post-increment
DMA5CONbits.MODE = 2;     // Configure DMA for Continuous Ping-Pong mode
DMA5PAD = (volatile unsigned int)&ADC1BUF0; // Point DMA to ADC1BUF0
DMA5CNT = 31;             // 32 DMA request
DMA5REQ = 13;             // Select ADC1 as DMA Request source

DMA5STAL = __builtin_dmaoffset(BufferA);
DMA5STAH = 0x0000;

DMA5STBL = __builtin_dmaoffset(BufferB);
DMA5STBH = 0x0000;

IFS3bits.DMA5IF = 0;      // Clear the DMA Interrupt Flag bit
IEC3bits.DMA5IE = 1;      // Set the DMA Interrupt Enable bit

DMA5CONbits.CHEN = 1;     // Enable DMA
```


© 2012 Microchip Technology Inc.

例 22-5: レジスタ間接アドレッシングによる ADC からのデータ転送 (続き)

DMA チャンネル 5 割り込みハンドラを設定する :

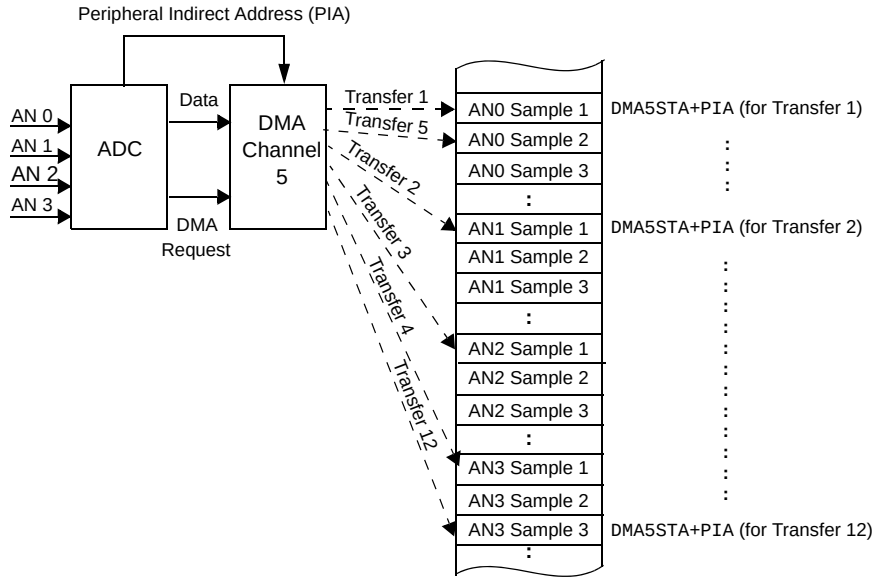
```
unsigned int DmaBuffer = 0;

void __attribute__((__interrupt__,no_auto_psv)) _DMA5Interrupt(void)
{
 // Switch between Primary and Secondary Ping-Pong buffers
if(DmaBuffer == 0)
 {
   ProcessADCSamples(BufferA);
 }
 else
 {
   ProcessADCSamples(BufferB);
 }

 DmaBuffer ^= 1;

 IFS3bits.DMA5IF = 0;    // Clear the DMA5 Interrupt Flag
}
```

ADC1 を DMA 動作用に設定する :

```
AD1CON1bits.ADDMABM = 0;  // Don't Care:ADC address generation is
                          // ignored by DMA
AD1CON2bits.SMPI    = 3;  // Don't Care
AD1CON4bits.DMABL   = 3;  // Don't Care

IFS0bits.AD1IF      = 0;  // Clear Analog-to-Digital Interrupt Flag bit
IEC0bits.AD1IE      = 0;  // Do Not Enable Analog-to-Digital interrupt
AD1CON1bits.ADON    = 1;  // Turn on the ADC
```

標準的なアルゴリズムでは、ADC チャンネルの変換順にデータを 1 つの連続したバッファ領域に格納するため、転送後にデータの並べ換えまたはインデックス処理による不要データ位置の読み飛ばしが必要です。これには追加の処理コードが必要であり、実行時間が増加します。ADC の周辺モジュール間接アドレッシング モードは、ADC チャンネル別のバッファにデータを格納する特殊なアドレッシング モードを定義します。DMA チャンネルを周辺モジュール間接アドレッシング モードに設定した場合、先の例とは異なり、DMA 転送された ADC データはチャンネル別のバッファに格納されます (図 22-13 参照)。

図 22-13: 周辺モジュール間接アドレッシングによる ADC からのデータ転送



この種の ADC アドレッシングを行う場合、ADCx 制御 1 レジスタ (ADxCON1) の DMA バッファ構築モードビット (ADDMABM) をクリアする必要があります。このビットがセットされている場合、ADC は変換順にアドレスを生成します (DMA ポストインクリメント アドレッシング モードを使うレジスタ間接と同じ)。

前述のように、周辺モジュール間接アドレッシングでは、DMA 開始アドレス レジスタ (DMAxSTA と DMAxSTB) をユーザ アプリケーションで初期化する際に、周辺モジュール間接アドレスに対応する下位ビットのビット数に特に注意が必要です。ADC の場合、このビット数は ADC バッファのサイズと個数によって決まります。

ADC バッファの個数は、ADCx 制御 2 レジスタ (ADxCON2) の DMA アドレス インクリメント レート ビット (SMPI<4:0>) の値を使って初期化されます。各 ADC バッファのサイズは、ADCx 制御4レジスタ(ADCxCON4)のアナログ入力あたりDMAバッファサイズ ビット(DMABL<2:0>) の値を使って初期化されます。例として SMPI<4:0> ビットと DMABL<2:0> ビットを共に「011」に初期化すると、8 ワード ($2^{DMABL<2:0>}$) の ADC バッファが 4 個 (SMPI<011:0> + 1) 配置されます (計 32 ワード = 64 バイト)。この場合、DMAxSTA と DMAxSTB に書き込まれるアドレスの下位 6 ビット ($2^{6\ bits}$ = 64 バイト) をゼロに設定する必要があります。

dsPIC DSC 向け MPLAB C コンパイラを使って DMAxSTA および DMAxSTAB レジスタを初期化する場合、データ属性で適切なデータ配置を指定する必要があります。上記条件の場合、例 22-6 に示すサンプルコードを使うと、DMAxSTA および DMAxSTB レジスタを適切に初期化できます。

例 22-6: dsPIC® DSC 向け MPLAB® C コンパイラによる DMA バッファの配置

```
__eds__ intBufferA[4][8] __attribute__((eds,aligned(64)));
__eds__ intBufferB[4][8] __attribute__((eds,aligned(64)));

DMA0STAL = __builtin_dmaoffset(BufferA);
DMA0STAH = 0x0000;

DMA0STBL = __builtin_dmaoffset(BufferB);
DMA0STBH = 0x0000;
```

例 22-7 に、このコンフィグレーション向けのサンプルコードを示します。

例 22-7: 周辺モジュール間接アドレッシングによる ADC データの DMA 転送

ADC1 をチャンネル 0 ～ 3 でサンプリングするように設定する :

```
AD1CON1bits.FORM = 3;  // Data Output Format:Signed Fraction (Q15 format)
AD1CON1bits.SSRC = 2;  // Sample Clock Source:GP timer starts conversion
AD1CON1bits.ASAM = 1;  // Sampling begins immediately after conversion
AD1CON1bits.AD12B = 0; // 10-bit ADC operation
AD1CON1bits.SIMSAM = 0; // Samples multiple channels sequentially

AD1CON2bits.BUFM = 0;
AD1CON2bits.CSCNA = 1; // Scan CH0+ Input Selections during Sample A bit
AD1CON2bits.CHPS = 0;  // Converts CH0

AD1CON3bits.ADRC = 0;  // ADC clock is derived from systems clock
AD1CON3bits.ADCS = 63; // ADC conversion clock

AD1CON4bits.ADDMAEN = 1;
//AD1CHS0:Analog-to-Digital Input Select Register
AD1CHS0bits.CH0SA = 0; // MUXA +ve input selection (AIN0) for CH0
AD1CHS0bits.CH0NA = 0; // MUXA -ve input selection (VREF-) for CH0

//AD1CHS123:Analog-to-Digital Input Select Register
AD1CHS123bits.CH123SA = 0;// MUXA +ve input selection (AIN0) for CH1
AD1CHS123bits.CH123NA = 0;// MUXA -ve input selection (VREF-) for CH1

//AD1CSSH/AD1CSSL:Analog-to-Digital Input Scan Selection Register
AD1CSSH = 0x0000;
AD1CSSL = 0x000F;      // Scan AIN0, AIN1, AIN2, AIN3 inputs
```

Timer3 を ADC1 の変換トリガ用に設定する :

```
TMR3 = 0x0000;
PR3 = 4999;// Trigger ADC1 every 125usec
IFS0bits.T3IF = 0;     // Clear Timer3 interrupt
IEC0bits.T3IE = 0;     // Disable Timer3 interrupt

T3CONbits.TON = 1;     // Start Timer3
```


© 2012 Microchip Technology Inc.

例 22-7: 周辺モジュール間接アドレッシングによる DMA データの DMA 転送 (続き)

DMA チャンネル 5 を周辺モジュール間接アドレッシング モードに設定する :

```
struct
{
 unsigned int Adc1Ch0[8];
 unsigned int Adc1Ch1[8];
 unsigned int Adc1Ch2[8];
 unsigned int Adc1Ch3[8];
} BufferA;

struct
{
 unsigned int Adc1Ch0[8];
 unsigned int Adc1Ch1[8];
 unsigned int Adc1Ch2[8];
 unsigned int Adc1Ch3[8];
} BufferB;

DMA5CONbits.AMODE = 2; // Configure DMA for Peripheral Indirect mode
DMA5CONbits.MODE = 2;  // Configure DMA for Continuous Ping-Pong mode
DMA5PAD = (volatile unsigned int)&ADC1BUF0; // Point DMA to ADC1BUF0
DMA5CNT = 31;          // 32 DMA request (4 buffers, each with 8 words)
DMA5REQ = 13;          // Select ADC1 as DMA request source

DMA5STAL = &BufferA;
DMA5STAH = 0x0000;

DMA5STBL = &BufferB;
DMA5STBH = 0x0000;

IFS3bits.DMA5IF = 0;   // Clear the DMA Interrupt Flag bit
IEC3bits.DMA5IE = 1;   // Set the DMA Interrupt Enable bit
DMA5CONbits.CHEN=1;    // Enable DMA
```

DMA チャンネル 5 割り込みハンドラを設定する :

```
unsigned int DmaBuffer = 0;

void __attribute__((__interrupt__,no_auto_psv)) _DMA5Interrupt(void)
{
 // Switch between Primary and Secondary Ping-Pong buffers
if(DmaBuffer == 0)
 {
   ProcessADCSamples(BufferA.Adc1Ch0);
   ProcessADCSamples(BufferA.Adc1Ch1);
   ProcessADCSamples(BufferA.Adc1Ch2);
   ProcessADCSamples(BufferA.Adc1Ch3);
 }
 else
 {
   ProcessADCSamples(BufferB.Adc1Ch0);
   ProcessADCSamples(BufferB.Adc1Ch1);
   ProcessADCSamples(BufferB.Adc1Ch2);
   ProcessADCSamples(BufferB.Adc1Ch3);
 }

 DmaBuffer ^= 1;

 IFS3bits.DMA5IF = 0;    // Clear the DMA5 Interrupt Flag
}
```

ADC1 を DMA 動作用に設定する :

```
AD1CON1bits.ADDMABM = 0; // DMA buffers are built in scatter/gather mode
AD1CON2bits.SMPI    = 3;// 4 ADC buffers
AD1CON4bits.DMABL   = 3;// Each buffer contains 8 words

IFS0bits.AD1IF      = 0;// Clear Analog-to-Digital Interrupt Flag bit
IEC0bits.AD1IE      = 0;// Do Not Enable Analog-to-Digital interrupt
AD1CON1bits.ADON    = 1;// Turn on the ADC
```

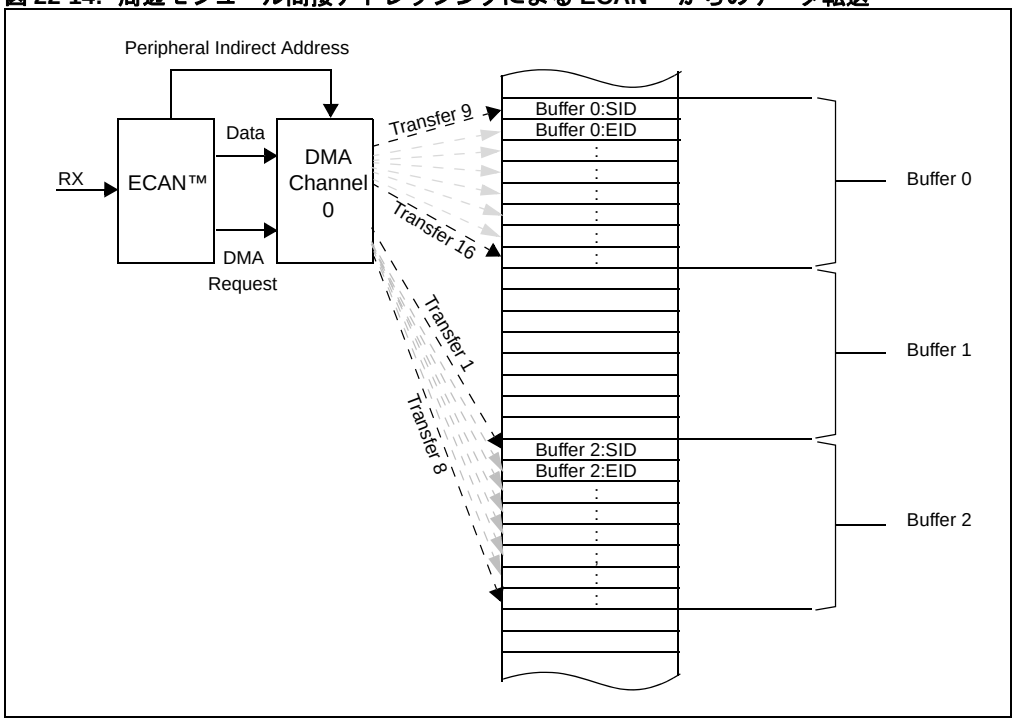
22.6.6.2 ECAN の DMA アドレス生成サポート

ECAN モジュールでも、周辺モジュール間接アドレッシング モードを使って特殊なアドレッシング機能を定義できます。dsPIC33E/PIC24E が CAN バス経由のメッセージをフィルタ処理して受信する場合、メッセージは下記の 2 つに分類できます。

- 処理する必要のある受信メッセージ
- 処理せずに他の CAN ノードへ転送する必要のある受信メッセージ

処理が必要な受信メッセージは、ユーザ アプリケーションで処理するために、8 ワードずつのバッファに再構成する必要があります。この場合、DPSRAM (または RAM) 内に複数の ECAN バッファを配置し、ECAN 周辺モジュール側で受信 (または送信) データのメモリアドレスを生成する事により、処理を単純化できます (図 22-14 参照)。この例では、受信データを最初にバッファ 2 に格納し、次にバッファ 0 に格納しています。ECAN モジュールは、メモリ内にデータを適切に格納するために、格納先アドレス(周辺モジュール間接アドレス)を生成します。

図 22-14: 周辺モジュール間接アドレッシングによる ECAN™ からのデータ転送



DMA が周辺モジュール間接アドレッシング モードで動作する場合、前述のように、DMA 開始アドレスレジスタ (DMAxSTA と DMAxSTB) をユーザ アプリケーションで初期化する際に、周辺モジュール間接アドレスに対応する下位ビットのビット数に特に注意が必要です。ECAN の場合、このビット数は、ECAN FIFO 制御レジスタ (CiFCTRL) の DMA バッファサイズ ビット (DMABS<2:0>) によって定義される ECAN バッファ数によって決まります。

例えば、ECAN モジュールが 12 個のバッファを確保 (DMABS<2:0> ビットを「011」に設定) している場合、12 個 x 8 ワード (計 96 ワード = 192 バイト) のバッファが配置されます。この場合、DMAxSTA および DMAxSTB レジスタに書き込まれるアドレスの下位 8 ビット (2^{8} bits = 256 バイト) を「0」に設定する必要があります。dsPIC DSC 向け MPLAB C コンパイラを使って DMAxSTA レジスタを初期化する場合、データ属性で適切なデータ配置を指定する必要があります。この例の場合、例 22-8 のサンプルコードを使うと、DMAxSTA レジスタを適切に初期化できます。

例 22-8: dsPIC® DSC 向け MPLAB® C コンパイラによる DMA バッファの配置

```
__eds__ intBufferA[12][8] __attribute__((eds,aligned(256)));

DMA0STAL = __builtin_dmaoffset(&BufferA[0][0]);
DMA0STAH = 0x0000;
```


© 2012 Microchip Technology Inc.

例 22-9 に、このコンフィグレーション向けのサンプルコードを示します。

受信メッセージを処理する必要がない場合もあります。例えば、車載アプリケーションでは、受信メッセージを CPU で処理せずに他のノードへ単純に転送する場合があります。この場合、受信バッファをメモリ内で並び換える必要はなく、転送可能となり次第、他のノードへ転送できます。

このようなデータ転送は、ポストインクリメント アドレッシング モードを使うレジスタ間接により DMA で実行できます。図 22-15 に、このプロセスを示します。

例 22-9: 周辺モジュール間接アドレッシングによる ECAN™ データの DMA 転送

2 つのフィルタを使って ECAN1 を設定する :

```
/* Initialize ECAN clock first.See ECAN section for example code */

C1CTRL1bits.WIN = 1;      // Enable filter window
C1FEN1bits.FLTEN0 = 1;    // Filter 0 is enabled
C1FEN1bits.FLTEN1 = 1;    // Filter 1 is enabled
C1BUFPNT1bits.F0BP = 0;   // Filter 0 points to Buffer0
C1BUFPNT1bits.F1BP = 2;   // Filter 1 points to Buffer2

C1RXF0SID = 0xFFEA;       // Filter 0 configuration
C1RXF0EID = 0xFFFF;

C1RXF1SID = 0xFFEB;       // Filter 1 configuration
C1RXF1EID = 0xFFFF;

C1FMSKSEL1bits.F0MSK = 0; // Mask 0 used for both filters
C1FMSKSEL1bits.F1MSK = 0; // Mask 0 used for both filters
C1RXM0SID = 0xFFEB;
C1RXM0EID = 0xFFFF;

C1FCTRLbits.DMABS = 3;    // 12 buffers in DMA RAM
C1FCTRLbits.FSA   = 3;    // FIFO starts from TX/RX Buffer 3

C1CTRL1bits.WIN     = 0;
C1TR01CONbits.TXEN0 = 0;  // Buffer 0 is a receive buffer
C1TR23CONbits.TXEN2 = 0;  // Buffer 2 is a receive buffer

C1TR01CONbits.TX0PRI = 0b11;// High Priority
C1TR01CONbits.TX1PRI = 0b10;// Intermediate High Priority

C1CTRL1bits.REQOP = 0;    // Enable Normal Operation mode
```

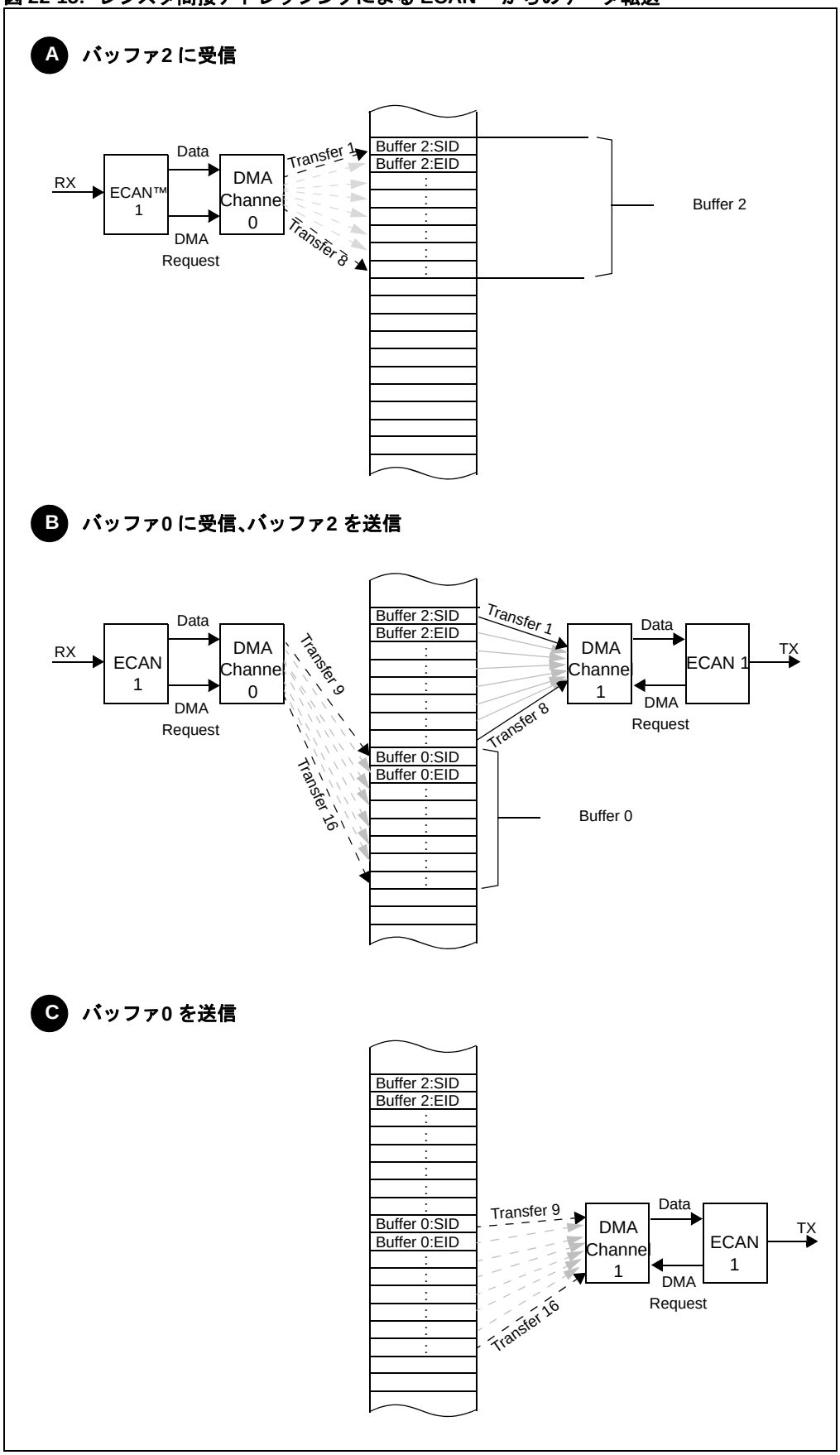
DMA チャンネル 0 を周辺モジュール間接アドレッシング モードに設定する :

```
__eds__Unsigned int Ecan1Rx[12][8] __attribute__((eds,space(dma)));
                          // 12 buffers,
                          // 8 words each

DMA0CONbits.AMODE = 2;    // Continuous mode, single buffer
DMA0CONbits.MODE = 0;     // Peripheral Indirect Addressing

DMA0PAD = (volatile unsigned int) &C1RXD;// Point to ECAN1 RX register

DMA0STAL = __builtin_dmaoffset(Ecan1Rx);
DMA0STAH = 0x0000;

DMA0CNT = 7;              // 8 DMA request (1 buffer, each with 8 words)
DMA0REQ = 0x22;           // Select ECAN1 RX as DMA Request source

IEC0bits.DMA0IE = 1;      // Enable DMA Channel 0 interrupt
DMA0CONbits.CHEN = 1;     // Enable DMA Channel 0
```

DMA 割り込みハンドラを設定する :

```
void __attribute__((__interrupt__)) _DMA0Interrupt(void)
{
   ProcessData(Ecan1Rx[C1VECbits.ICODE]);// Process received buffer;

   IFS0bits.DMA0IF = 0;   // Clear the DMA0 Interrupt Flag;
}
```

図 22-15: レジスタ間接アドレッシングによる ECAN™ からのデータ転送

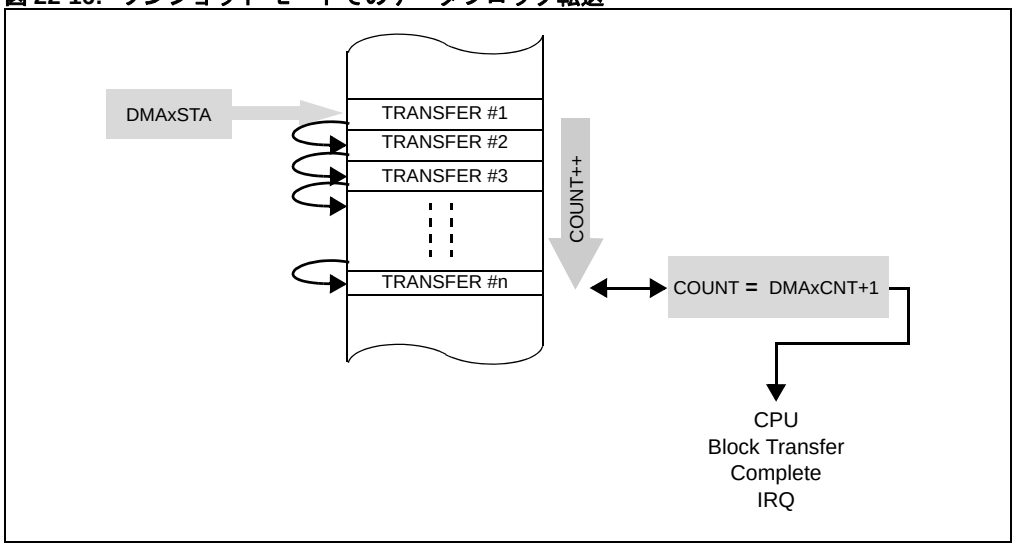


© 2012 Microchip Technology Inc.

22.6.7 ワンショット モード

データ転送を繰り返す必要がない場合、アプリケーション プログラムはワンショット モードを使います。DMA チャンネル制御レジスタ (DMAxCON) の動作モード選択ビットビット (MODE<1:0>) を「x1」に設定すると、このモードが選択されます。このモードでは、データブロック (ブロック長は DMAxCNT で定義) が完全に転送されてブロック終端が検出された時点で、そのチャンネルは自動的に無効になります (ハードウェアが DMA チャンネル制御レジスタ (DMAxCON) の CHEN ビットがクリアします)。図 22-16 にワンショット モードの動作を示します。

図 22-16: ワンショット モードでのデータ ブロック転送



DMA チャンネル制御レジスタ (DMAxCON) の HALF ビットがセットされている場合、データブロックの半分 (ハーフブロック) が転送された時点で DMAxIF ビットがセットされ、DMA 割り込みが生成されます (その割り込みがアプリケーション プログラムで有効にされている場合のみ)。この時点では、チャンネルは無効になりません。ブロック全体 (フルブロック) の転送が完了するとチャンネルは自動的に無効になります (割り込みフラグはセットされない)。DMA チャンネルをハーフブロック転送とフルブロック転送の両方で割り込むように設定する方法は、**22.6.3「フルブロックまたはハーフブロック転送割り込み」**を参照してください。

DMAxCON レジスタの CHEN ビットを「1」にセットしてそのチャンネルを再度有効にすると、DMA 開始アドレスレジスタ (DMAxSTA、DMAxSTB) が指定する開始アドレスからブロック転送が始まります。例 22-10 に、ワンショット動作向けのサンプルコードを示します。

例 22-10:ワンショット モードによる UART データの DMA 転送

UART を RX および TX 用に設定する :

```
#define FCY      40000000
#define BAUDRATE 9600
#define BRGVAL   ((FCY/BAUDRATE)/16) - 1

U2MODEbits.STSEL = 0;  // 1-stop bit
U2MODEbits.PDSEL = 0;  // No parity, 8-data bits
U2MODEbits.ABAUD = 0;  // Auto-baud disabled

U2BRG = BRGVAL;        // Baud rate setting for 9600

U2STAbits.UTXISEL0 = 0;// Interrupt after one TX character is transmitted
U2STAbits.UTXISEL1 = 0;
U2STAbits.URXISEL = 0; // Interrupt after one RX character is received

U2MODEbits.UARTEN   = 1;// Enable UART
U2STAbits.UTXEN     = 1;// Enable UART TX
```

例 22-10:ワンショット モードによる UART データの DMA 転送 (続き)

DMA チャンネル 0 をワンショット / シングルバッファ モードの送信用に設定する :

```
unsigned int BufferA[8];
unsigned int BufferB[8];

DMA0CON = 0x2001;      // One-Shot, Post-Increment, RAM-to-Peripheral
DMA0CNT = 7;           // Eight DMA requests
DMA0REQ = 0x001F;      // Select UART2 transmitter

DMA0PAD = (volatile unsigned int) &U2TXREG;

DMA0STAL = __builtin_dmaoffset(BufferA);
DMA0STAH = 0x0000;

IFS0bits.DMA0IF = 0;   // Clear DMA Interrupt Flag
IEC0bits.DMA0IE = 1;   // Enable DMA interrupt
```

DMA チャンネル 1 を連続 / ピンポンモードの受信用に設定する :

```
DMA1CON = 0x0002;      // Continuous, Ping-Pong, Post-Inc., Periph-RAM
DMA1CNT = 7;           // Eight DMA requests
DMA1REQ = 0x001E;      // Select UART2 receiver

DMA1PAD = (volatile unsigned int) &U2RXREG;

DMA1STAL = &BufferA;
DMA1STAH = 0x0000;

DMA1STBL = &BufferB;
DMA1STBH = 0x0000;

IFS0bits.DMA1IF = 0;   // Clear DMA interrupt
IEC0bits.DMA1IE = 1;   // Enable DMA interrupt
DMA1CONbits.CHEN = 1;  // Enable DMA channel
```

DMA 割り込みハンドラを設定する :

```
void __attribute__((__interrupt__,no_auto_psv)) _DMA0Interrupt(void)
{
   IFS0bits.DMA0IF = 0;// Clear the DMA0 Interrupt Flag
}


void __attribute__((__interrupt__,no_auto_psv)) _DMA1Interrupt(void)
{
   static unsigned int BufferCount = 0; // Keep record of which buffer
                                        // contains RX Data

 if(BufferCount == 0)
   {
 DMA0STAL = __builtin_dmaoffset(BufferA);
 DMA0STAH = 0x0000;
 }
 else
{
 DMA0STBL = __builtin_dmaoffset(BufferB);
 DMA0STBH = 0x0000;
 }
 DMA0CONbits.CHEN = 1;   // Enable DMA0 channel
 DMA0REQbits.FORCE = 1;  // Manual mode:Kick-start the 1st transfer

 BufferCount ^= 1;
 IFS0bits.DMA1IF = 0;    // Clear the DMA1 Interrupt Flag
}
```

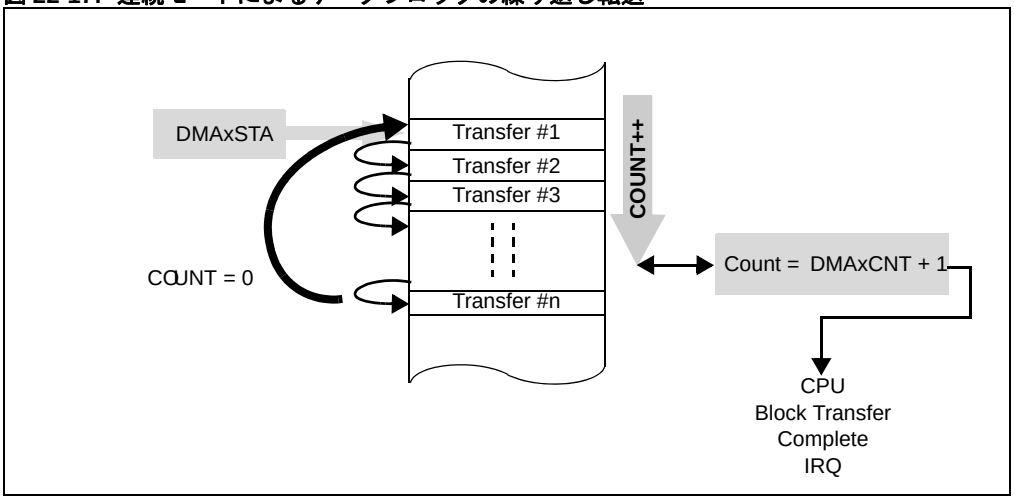

© 2012 Microchip Technology Inc.

22.6.8 連続モード

プログラムが動作している間転送を繰り返し実行する必要がある場合、アプリケーション プログラムは連続モードを使います。

DMA チャンネル制御レジスタ (DMAxCON) の動作モード選択ビットビット (MODE<1:0>) を「x0」に設定すると、このモードが選択されます。このモードでは、データブロック (ブロック長は DMAxCNT で定義) が完全に転送されてブロック終端が検出されてもチャンネルは無効になりません。ブロックの最後のデータの転送中に、DMA DPSRAM/RAM アドレスは (プライマリ)DMA 開始アドレス A レジスタ (DMAxSTA) へリセットされます。図 22-17 に連続モードの動作を示します。

図 22-17: 連続モードによるデータブロックの繰り返し転送



DMA チャンネル制御レジスタ (DMAxCON) の HALF ビットがセットされている場合、データブロックの半分 (ハーフブロック) が転送された時点で DMAxIF ビットがセットされ、DMA 割り込みが生成されます (その割り込みがアプリケーション プログラムで有効にされている場合のみ)。この時、チャンネルは無効になりません。ブロック全体 (フルブロック) の転送が完了しても、チャンネルは無効になりません (割り込みフラグはセットされない)。DMA チャンネルをハーフブロック転送とフルブロック転送の両方で割り込むように設定する方法は、**22.6.3「フルブロックまたはハーフブロック転送割り込み」**を参照してください。

22.6.9 ピンポンモード

ピンポンモードでは 2 つのバッファを使用し、DMA チャンネルが一方のバッファを使っている間に、CPU は他方のバッファを処理できます。この結果、CPU は DMA チャンネルが一方のバッファで DMA ブロック転送を完了するまでの時間を使って他方のバッファのデータを処理できます。当然ですが、この転送モードは所定サイズのバッファリングに他のモードの 2 倍のメモリを使います。

全ての DMA 動作モードでは、DMA チャンネルを有効にした時に、既定値により DMA チャンネル x メモリ開始アドレス A レジスタ (DMAxSTA) を使って初期のメモリ実効アドレスを生成します。1 ブロックの転送が完了して DMA チャンネルが再初期化されるたびに、バッファ開始アドレスは同じ DMAxSTA レジスタから読み出されます。

これに対しピンポンモードでは、バッファ開始アドレスは下記の 2 つのレジスタから読み出されます。

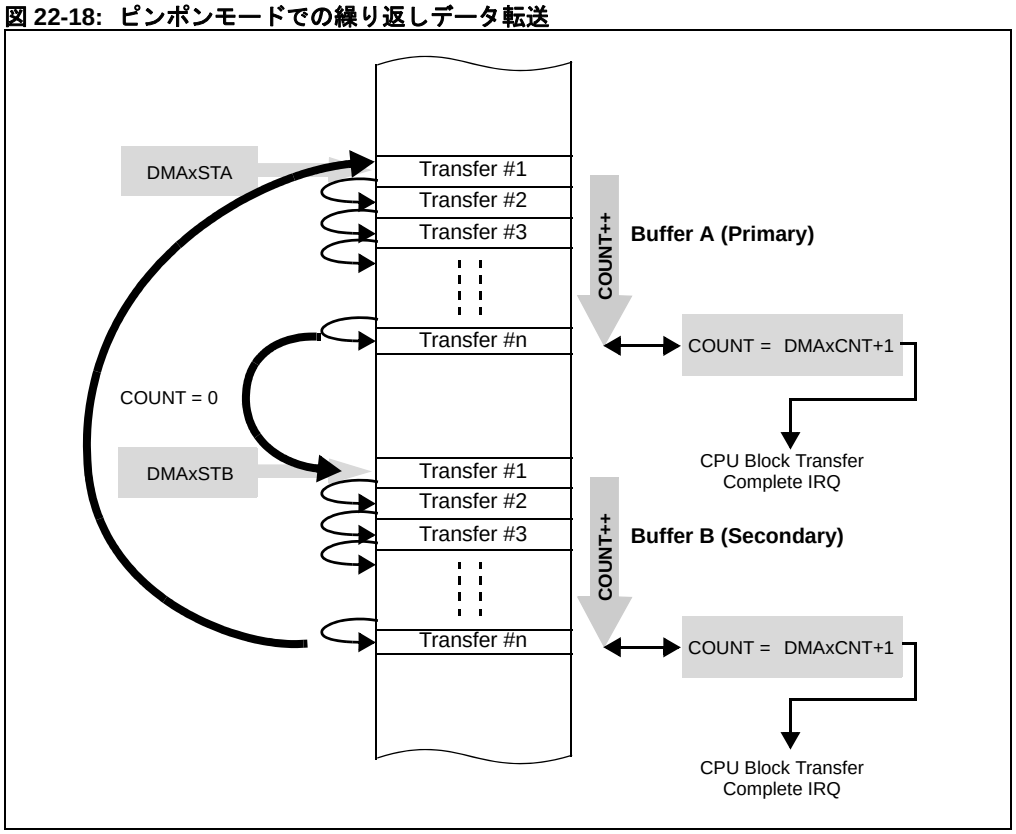
- プライマリ : DMA チャンネル x 開始アドレス A レジスタ (DMAxSTA)
- セカンダリ : DMA チャンネル x 開始アドレス B レジスタ (DMAxSTB)

DMA は、プライマリ バッファとセカンダリ バッファをブロック転送ごとに交互に使います。1 ブロックの転送が完了して DMA チャンネルが再初期化されるたびに、バッファ開始アドレスは 2 つのレジスタの一方から交互に読み出されます。DMA ピンポン ステータス レジスタ (DMAPPS) の PPSTx ビットは、DMA コントローラが新しいブロックの最初のバイト / ワードを転送した後に更新されます。

DMA チャンネル制御レジスタ (DMAxCON) の動作モード選択ビットビット (MODE<1:0>) を「1x」に設定すると、ピンポンモードが選択されます。

連続モードの DMA でピンポンモードを有効にすると、DMA はプライマリ バッファを使って転送した後、再初期化時にセカンダリ バッファを選択して次の転送を実行します。後続のブロック転送には、プライマリ バッファとセカンダリ バッファを交互に使います。各バッファの転送後に割り込みが生成されます (割り込みがアプリケーション プログラムで有効にされている場合のみ)。図 22-18 に、ピンポンモード有効時の連続モード動作を示します。例 22-11 に、DCI モジュールを使ったピンポンモード動作用のサンプルコードを示します。

図 22-18: ピンポンモードでの繰り返しデータ転送



© 2012 Microchip Technology Inc.

例 22-11:連続ピンポンモードによる DCI データの DMA 転送

DCI を RX および TX 用に設定する :

```
#define FCY 40000000
#define FS  48000
#define FCSCK64*FS
#define BCGVAL(FCY/(2*FS))-1

DCICON1bits.CSCKD = 0;// Serial Bit Clock (CSCK pin) is output
DCICON1bits.CSCKE = 0;// Data sampled on falling edge of CSCK
DCICON1bits.COFSD = 0;// Frame Sync Signal is output
DCICON1bits.UNFM = 0;// Transmit ‘0’s on a transmit underflow
DCICON1bits.CSDOM = 0;// CSDO pin drives ‘0’s during disabled TX time slots
DCICON1bits.DJST = 0;// TX/RX starts 1 serial clock cycle after frame sync
pulse
DCICON1bits.COFSM = 1;// Frame Sync signal set up for I2S mode

DCICON2bits.BLEN = 0;// One data word will be buffered between interrupts
DCICON2bits.COFSG = 1;// Data frame has 2 words:LEFT & RIGHT samples
DCICON2bits.WS = 15;// Data word size is 16 bits

DCICON3 = BCG_VAL; // Set up CSCK Bit Clock Frequency

TSCONbits.TSE0 = 1;    // Transmit on Time Slot 0
TSCONbits.TSE1 = 1;    // Transmit on Time Slot 1
TSCONbits.TSE0 = 1;    // Transmit on Time Slot 0
RSCONbits.RSE1 = 1;    // Receive on Time Slot 1
```

DMA チャンネル 0 を連続ピンポンモードでの送信用に設定する :

```
__eds__ unsigned int TxBufferA[16] __attribute__((eds,space(dma)));
__eds__ unsigned int TxBufferB[16] __attribute__((eds,space(dma)));

DMA0CON = 0x2002;  // Ping-Pong, Continuous, Post-Increment,
RAM-to-Peripheral
DMA0CNT = 15;      // 15 DMA requests
DMA0REQ = 0x003C;  // Select DCI as DMA Request source

DMA0PAD = (volatile unsigned int) &TXBUF0;

DMA0STAL = __builtin_dmaoffset(TxBufferA);
DMA0STAH = 0x0000;

DMA0STBL = __builtin_dmaoffset(TxBufferB);
DMA0STBH = 0x0000;

IFS0bits.DMA0IF = 0;// Clear DMA Interrupt Flag
IEC0bits.DMA0IE = 1;// Enable DMA interrupt
DMA0CONbits.CHEN = 1;// Enable DMA channel
```

DMA チャンネル 1 を連続ピンポンモードでの受信用に設定する :

```
__eds__ unsigned int RxBufferA[16] __attribute__((eds,space(dma)));
__eds__ unsigned int RxBufferB[16] __attribute__((eds,space(dma)));

DMA1CON = 0x0002;  // Continuous, Ping-Pong, Post-Inc., Periph-RAM
DMA1CNT = 15;      // 16 DMA requests
DMA1REQ = 0x003C;  // Select DCI as DMA Request source

DMA1PAD = (volatile unsigned int) &RXBUF0;

DMA1STAL = __builtin_dmaoffset(RxBufferA);
DMA1STAH = 0x0000;

DMA1STBL = __builtin_dmaoffset(RxBufferB);
DMA1STBH = 0x0000;

IFS0bits.DMA1IF = 0;// Clear DMA interrupt
IEC0bits.DMA1IE = 1;// Enable DMA interrupt
```

例 22-11: 連続ピンポンモードによる DCI データの DMA 転送 (続き)

DMA 割り込みハンドラを設定する :

```
void __attribute__((__interrupt__,no_auto_psv)) _DMA0Interrupt(void)
{
 static unsigned int TxBufferCount = 0;// Keep record of which buffer
                                       // has RX Data

   if(BufferCount == 0)
   {
 /* Notify application that TxBufferA has been transmitted */
 }
 else
 {
 /* Notify application that TxBufferB has been transmitted */
 }

 BufferCount ^= 1;
 IFS0bits.DMA0IF = 0;             // Clear the DMA0 Interrupt Flag
}

void __attribute__((__interrupt__,no_auto_psv)) _DMA1Interrupt(void)
{
   static unsigned int RxBufferCount = 0; // Keep record of which buffer
                                    //  has RX Data

 if(BufferCount == 0)
 {
 /* Notify application that RxBufferA has been received */
 }
 else
 {
 /* Notify application that RxBufferB has been received */
 }

 BufferCount ^= 1;
 IFS0bits.DMA1IF = 0;             // Clear the DMA1 Interrupt Flag
}
```

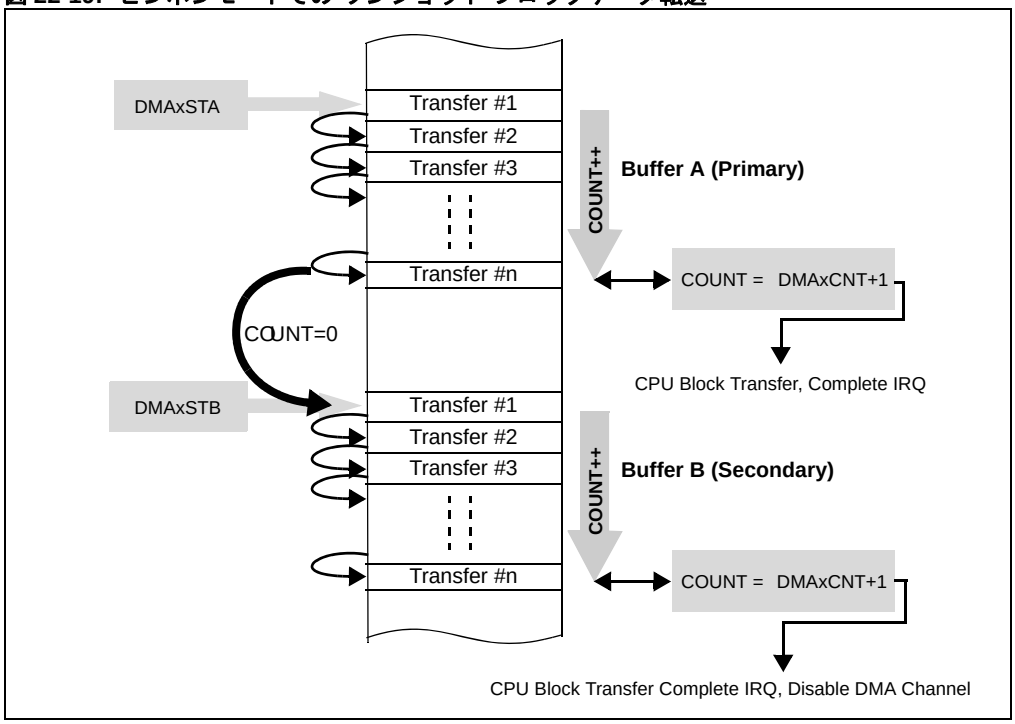
DCI を有効にする :

```
/* Force first two words to fill-in TX buffer/shift register */
DMA0REQbits.FORCE = 1;
while(DMA0REQbits.FORCE == 1);

DMA0REQbits.FORCE = 1;
while(DMA0REQbits.FORCE == 1);

DCICON1bits.DCIEN = 1;            // Enable DCI
```

ピンポンモードの DMA でワンショット モードを有効にすると、DMA はプライマリ バッファを使って転送した後、再初期化時にセカンダリ バッファを選択して次の転送を実行します。その後 DMA チャンネルは無効になるため、後続の転送は発生しません。図 22-19 に、ピンポンモード有効時のワンショット データ転送の動作を示します。

© 2012 Microchip Technology Inc.

図 22-19: ピンポンモードでの ワンショット ブロックデータ転送



22.6.10 手動転送モード

DMA コントローラを使って周辺モジュールからメモリにデータを送信する場合、DMA 転送は DMA チャンネルと周辺モジュールの初期化後自動的に始まります。周辺モジュールは、メモリへのデータ転送が可能になると DMA 要求を生成します。この時にメモリから周辺モジュールへもデータを送信する必要がある場合、同じ DMA 要求を使ってもう 1 チャンネルを有効にする事により、メモリからデータを読み出して周辺モジュールに書き込む事ができます。

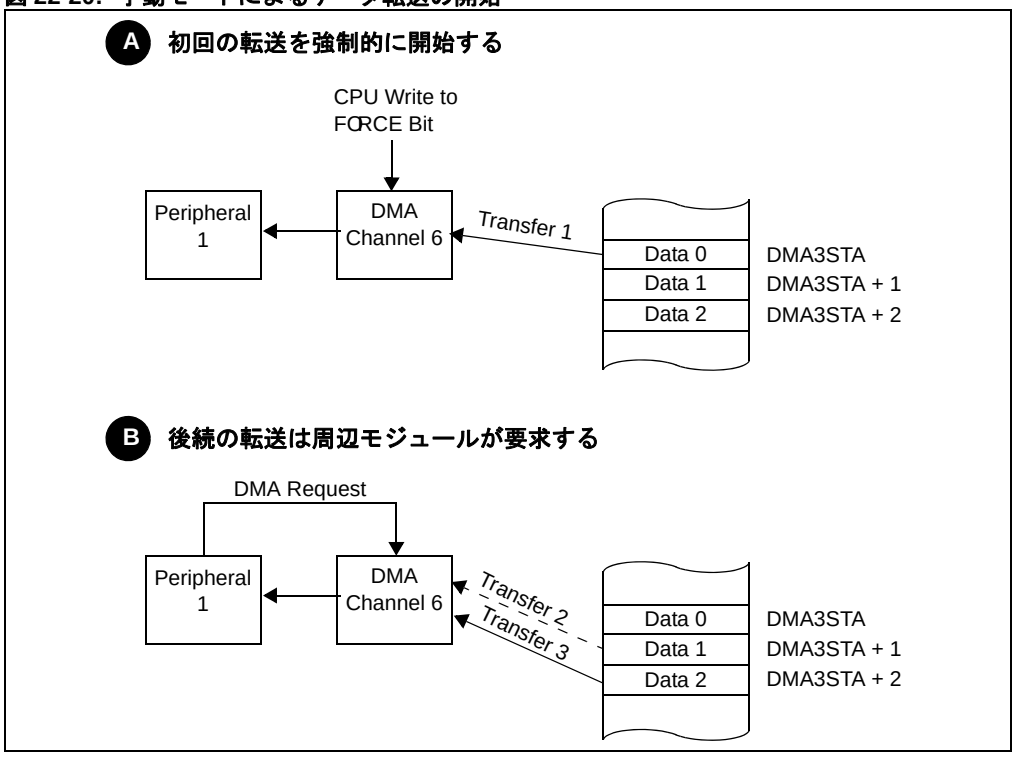
メモリバッファから周辺モジュールへの単方向のデータ送信だけが必要な場合、処理を開始するには、最初のデータを手動で周辺モジュールに転送する必要があります (**22.7「DMA 転送の開始」**参照)。このプロセスはソフトウェアで開始できますが、より便利な方法として、DMA チャンネル レジスタ内の 1 ビットをセットするだけで、そのチャンネルの DMA 要求を簡単に模倣できます。DMA チャンネルは、模倣された DMA 要求を通常の要求と同様に処理し、最初のデータ要素を転送してシーケンスを開始します。周辺モジュールは、次のデータの受信準備が完了した時に通常の DMA 要求を送信し、これを受けた DMA は次のデータ要素を送信します。このプロセスを図 22-20 に示します。

手動 DMA 要求は、DMA チャンネル x IRQ 選択レジスタ (DMAxREQ) の FORCE ビットをセットする事により生成できます。1 度セットした FORCE ビットをユーザ アプリケーションでクリアする事はできません。FORCE ビットは、手動 DMA 転送が完了した時にハードウェアによってクリアされます。FORCE ビットをセットするタイミングによっては、下記の特殊な条件が適用されます。

- DMA 転送実行中に FORCE ビットをセットしても何も効果を持たず無視されます。
- チャンネル x の設定時に FORCE ビットをセットした場合 (DMA チャンネルを設定するためのレジスタへの書き込みで同時に FORCE ビットもセットした場合)、予測不可能な挙動が生じます。従って、このような操作を避ける必要があります。
- FORCE ビットをセットすると、DMA 転送カウントは 1 つデクリメントします。この事は、DMA を使って転送されたデータの最初のブロックを処理する際に、ユーザ側のソフトウェアで考慮する必要があります。
- DMA チャンネルの周辺モジュール割り込み要求が保留中である場合、そのチャンネルの FORCE ビットをセットしても無視され、割り込みによる要求が優先されます。ただし、DMA 要求コリジョン ステータス レジスタ (DMARQC) のチャンネル x コリジョン フラグビット (RQCOLx) がセットされてエラー条件が生成されます。詳細は **22.10「データ書き込みコリジョンと要求コリジョン」**を参照してください。

Note:	DMA チャンネルを設定するための書き込みで同時に FORCE ビットをセットしないでください。これは予測不可能な挙動を生じる可能性があります。

図 22-20: 手動モードによるデータ転送の開始



22.6.11 NULL データ書き込みモード

NULL データ書き込みモードは、SPI のようにデータを一切送信せずにシーケンシャルなデータ受信だけを必要とするアプリケーションで非常に便利です。

基本的に SPI は単純なシフトレジスタであり、1 クロック周期あたり 1 ビットのデータを出し入れします。受信データだけが必要であるにも関わらず SPI をマスタモード (SPI がクロックの供給源) に設定すると、特殊な状況が発生します。すなわち、SPI データクロックを起動して外部データを受信するには、SPI データレジスタに何らかのデータを書き込む必要があります。

2 つの DMA チャンネルを割り当てて、1 チャンネルをデータ受信用に使い、もう 1 チャンネルを NULL またはゼロデータを SPI に送信するためだけに使う方法もあります。しかし、DMA NULL データ書き込みモードを使うと、より簡単に NULL データを送信できます。このモードでは、周辺モジュール データの読み出し用に設定した DMA チャンネルがデータ要素を受信および転送するたびに、SPI データレジスタに NULL 値が自動的に書き込まれます。

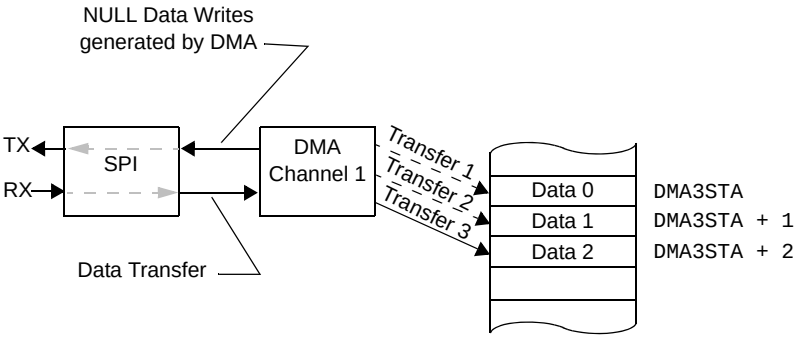
DMA チャンネル x 制御レジスタ (DMAxCON) の NULL データ周辺モジュール書き込みモード選択ビット (NULLW) がセットされ、かつ、DMA チャンネルが周辺モジュールからの読み出し用に設定されている場合、DMA チャンネルは周辺モジュールデータの読み出しサイクル中に NULL データ (全ゼロデータ) を周辺モジュール アドレスに書き込みます。この書き込みは周辺モジュールバスを使い、DPSRAM/RAM へのデータ書き込みと同時に発生します。図 22-21 に NULL データ書き込みモードの動作を示します。

通常このモードでは、周辺モジュール DMA 要求への応答時 (データを受信して転送可能となった時) にのみ NULL データ書き込みが発生します。最初のワードの受信を開始するために、CPU から周辺モジュールへの初期書き込みが必要です。その後、DMA は後続の全ての周辺モジュール (NULL) データ書き込みを処理します。つまり、CPU による NULL 書き込みによって SPI (マスタ) のデータ送受信が始まり、その結果、新たに受信したデータを転送するために DMA 要求が生成されます。

手動 DMA 転送を使ってこのプロセスを開始する事もできます。しかしこの方法では、冗長な周辺モジュール読み出し (データは無効) と関連メモリポインタの調整が必要である事に注意が必要です。

© 2012 Microchip Technology Inc.

図 22-21: NULL データ書き込みモードによるデータ転送



例 22-12:NULL データ書き込みモードによる SPI データの DMA 転送

SPI をマスタモードに設定する :

```
SPI1CON1bits.MODE16 = 1;  //Communication is word-wide (16 bits)
SPI1CON1bits.MSTEN = 1;   //Master mode enabled
SPI1STATbits.SPIEN = 1;   //Enable SPI module
```

DMA チャンネル 1 を NULL データ書き込みモードに設定する :

```
unsigned int BufferA[16];
unsigned int BufferB[16];

DMA1CON = 0x0802;         // Null Write, Continuous, Ping-Pong,
                          // Post-Increment, Periph-to-RAM
DMA1CNT = 15;             // Transfer 16 words at a time
DMA1REQ = 0x000A;         // Select SPI1 as DMA Request source

DMA1STAL = &BufferA;      // The buffer is in non-DMA RAM area
DMA1STAH = 0x0000;

DMA1STBL = &BufferB;      // The buffer is in non-DMA RAM area
DMA1STBH = 0x0000;

DMA1PAD = (volatile unsigned int) &SPI1BUF;

IFS0bits.DMA1IF = 0;
IEC0bits.DMA1IE = 1;      // Enable DMA interrupt
DMA1CONbits.CHEN = 1;     // Enable DMA channel

DMA1REQbits.FORCE = 1;    // Force first word after enabling SPI
```

DMA 割り込みハンドラを設定する :

```
void __attribute__((__interrupt__,no_auto_psv)) _DMA1Interrupt(void)
{
   static unsigned int BufferCount = 0;// Keep record of which buffer
                                       // contains RX data

 if(BufferCount == 0)
 {
 ProcessRxData(BufferA);          // Process received SPI data in
                                  // DMA RAM Primary buffer
 }
 else
 {
 ProcessRxData(BufferB);          // Process received SPI data in
                                  // DMA RAM Secondary buffer
 }

 BufferCount ^= 1;
 IFS0bits.DMA1IF = 0;             // Clear the DMA1 Interrupt Flag
}
```

22.7 DMA 転送の開始

DMA 転送を開始する前に、DMAxCON レジスタの CHEN ビットを「1」にセットして、そのDMA チャンネルを有効にする必要があります。DMA チャンネルが既に有効である場合、そのチャンネルをまず無効 (CHEN = 0) にしてから再度有効 (CHEN = 1) にする事によって、そのチャンネルを再初期化できます。この操作により、DMA 転送カウントはゼロにリセットされ、DMA バッファにはプライマリバッファが選択されます。

DMA チャンネルと周辺モジュールが正しく初期化されていれば、周辺モジュールがデータ転送の準備を完了して DMA 要求を生成した時点で DMA 転送が始まります。ただし、一部の周辺モジュールは、一定の条件が成立するまで DMA 要求を生成しません (従って DMA 転送は始まりません)。このような場合、DMA 転送を開始するには、各種の DMA モードとプロセスの組み合わせを適用する必要があります。

22.7.1 シリアル ペリフェラル インターフェイス (SPI) との DMA 転送の開始

SPI 周辺モジュールとの DMA 転送の開始方法は、SPI データの転送方向とスレーブ / マスタのどちらのモードで転送するかによって異なります。

- **マスタモードで TX のみ** – このコンフィグレーションでは、SPI データの先頭ブロックを送信するまで DMA 要求は発行されません。DMA 転送を開始するには、ユーザ アプリケーションで DMA 手動転送モードを使って最初にデータを送信するか、あるいは DMA を使わず最初に SPI バッファ (SPIxBUF) にデータを書き込む必要があります。
- **マスタモードで RX のみ** – このコンフィグレーションでは、SPI データの先頭ブロックを受信するまで DMA 要求は発行されません。しかしマスタモードでは、SPI 自身が最初に送信するまでデータは受信されません。DMA 転送を開始するには、ユーザ アプリケーションで DMA NULL データ書き込みモードを使って DMA 手動転送モードを開始する必要があります。
- **マスタモードで RX と TX** – このコンフィグレーションでは、SPI データの先頭ブロックを受信するまで DMA 要求は発行されません。しかしマスタモードでは SPI 自身が最初に送信するまでデータは受信されません。DMA 転送を開始するには、ユーザ アプリケーションで DMA 手動転送モードを使って最初にデータを送信するか、あるいは DMA を使わず最初に SPI バッファ (SPIxBUF) にデータを書き込む必要があります。
- **スレーブモードで RX のみ** – このコンフィグレーションでは、SPI データの先頭ブロックを受信するまで DMA 要求は発行されません。DMA を開始するにはユーザ アプリケーションで DMA 手動転送モードを使って最初にデータを送信するか、あるいは DMA を使わず最初に SPI バッファ (SPIxBUF) にデータを書き込む必要があります。
- **スレーブモードで RX のみ** – このコンフィグレーションでは、最初の SPI データが到着すると即座に DMA 要求が生成されます。従って、DMA 転送を開始するための特別な手順は不要です。
- **スレーブモードで RX と TX** – このコンフィグレーションでは、SPI データの先頭ブロックを受信するまで DMA 要求は発行されません。DMA 転送を開始するには、ユーザ アプリケーションで DMA 手動転送モードを使って最初にデータを送信するか、あるいは DMA を使わず最初に SPI バッファ (SPIxBUF) にデータを書き込む必要があります。

22.7.2 データコンバータ インターフェイス (DCI) との DMA 転送の開始

他のシリアル ペリフェラルとは異なり、DCI は有効になると即座に転送を始めます (DCI がマスタである場合)。DCI は、接続先の外部コーデックに向けてデータの同期フレームを常時供給します。DCI を有効にする前に、ユーザは下記を行う必要があります。

- **22.5.2「周辺モジュールのコンフィグレーション」**の説明に従って DCI を設定します。
- ステレオ コーデックに接続する場合、最初の 2 個のデータの転送を開始するために、下記のように DMA 手動転送モードを使います。
  - DMAxREQ レジスタの FORCE ビットをセットして DCI の左チャンネル サンプルを転送する
  - 再度 FORCE ビットをセットして DCI の右チャンネル サンプルを転送する

上記を実行した後に、DCI 周辺モジュールを有効にします (例 22-11 参照)。

22.7.3 UART との DMA 転送の開始

UART レシーバは、データを受信すると即座に DMA 要求を生成します。DMA 転送を開始するためにユーザ アプリケーションによる特別な手順は不要です。UART とトランスミッタが有効になり次第、UART トランスミッタは DMA 要求を発行します。従って、UART とトランスミッタを有効にする前に、DMA チャンネルとバッファを初期化し有効にしておく必要があります。UART は、表 22-2 に従って設定する必要があります。

あるいは、DMA チャンネルを有効にする前に、UART と UART トランスミッタを有効にする事もできます。この場合、UART トランスミッタの DMA 要求は失効するため、DMA 転送を開始するためにユーザ アプリケーションで DMAxREQ レジスタの FORCE ビットをセットして DMA 要求を発行する必要があります。

© 2012 Microchip Technology Inc.

22.8 DMA チャンネルの調停とオーバーラン

各 DMA チャンネルの優先度は、チャンネル番号順に固定されています (チャンネル 0 の優先度が最高、チャンネル 7 の優先度が最低)。DMA 転送要求は、その要求源に割り当てられた DMA チャンネルによってラッチされます。DMA コントローラはアービトレータとして動作します。実行中または保留中の転送が他に存在しない場合、コントローラは要求中の DMA チャンネルに対してバスリソースの使用を承認します。DMA コントローラは、バスリソースを使用中の DMA チャンネルが動作を完了するまで、そのリソースを他の DMA チャンネルに一切使用させません。

複数の DMA 要求を同時に受け取った場合、または複数の DMA 要求が保留中である場合、DMA コントローラの優先度ロジックは、その中で最も優先度の高い DMA チャンネルにリソースの使用を承認します。他の全ての DMA 要求は、最優先の DMA 転送が完了するまで保留されます。DMA 転送の実行中に新たな DMA 要求が届いた場合、その要求も既に保留中の DMA 要求に加えた上で優先度が再評価されます。これにより、常に最優先の要求が実行中の DMA 転送完了後に処理されます。

DMA チャンネルが DPSRAM 領域外のメモリにアクセスを試み、メモリアービターがアクセスを許可しない場合、DMA チャンネルはアービターが要求を承認してアクセスを許可するまでストールします。結果として、後続の全ての DMA チャンネル要求は、ストールしたチャンネルに対する優先度の優劣に関係なく、アービターが未解決の要求を解決するまで保留される事に注意してください。

このように、DMA チャンネルは優先度に基づいて処理されるため、DMA 要求は即座に処理されず保留される可能性があります。優先度の低い要求は、より高優先度のチャンネルが全て処理されるまで保留されます。DMA コントローラが先に受け取った DMA 要求をクリアする前に別の割り込みを受け取り、その割り込みが保留中の割り込みと同タイプであった場合、データオーバーランが発生します。

データ オーバーランは、DMA が先のデータを移動し終える前に、周辺モジュール データバッファに新たなデータが届いた状態として定義されます。一部の DMA 対応周辺モジュールは、データ オーバーランを検出して CPU 割り込みを生成できます (その周辺モジュール エラー割り込みが有効になっている場合のみ)。表 22-3 を参照してください。

表 22-3: DMA 対応周辺モジュールによるオーバーラン処理

DMA 対応周辺モジュール	データ オーバーラン処理
シリアル ペリフェラル インターフェイス (SPI)	DMA チャンネルがまだ移動し終えていないデータは、後続の受信データによって上書きされません。後続の受信データは破棄され、SPI ステータスレジスタ (SPIxSTAT) の SPI 受信オーバーフロービット (SPIROV) がセットされます。加えて、割り込みコントローラで割り込みイネーブル制御レジスタ (IECx) の SPI エラー割り込みイネーブルビット (SPIxEIE) がセットされていれば、SPIx フォルト割り込みが生成されます。
UART (Universal Asynchronous Receiver Transmitter)	DMA チャンネルがまだ移動し終えていないデータは、後続の受信データによって上書きされません。後続の受信データは破棄され、UART ステータス レジスタ (UxSTA) のオーバーフロー エラービット (OERR) がセットされます。加えて、割り込みコントローラで割り込みイネーブル制御レジスタ (IECx) の UART エラー割り込みイネーブルビット (UxEIE) がセットされていれば、UARTx エラー割り込みが生成されます。
データコンバータ インターフェイス (DCI)	DMA チャンネルがまだ移動し終えていないデータは、後続の受信データによって上書きされ、DCI ステータスレジスタ (DCISTAT) の受信オーバーフロービット (ROV) がセットされます。加えて、割り込みコントローラで割り込みイネーブル制御レジスタ (IEC0) の DCI エラー割り込みイネーブルビット (DCIEIE) がセットされていれば、DCI フォルト割り込みが生成されます。
10/12 ビット A/D コンバータ (ADC)	DMA チャンネルがまだ移動し終えていないデータは、後続の受信データによって上書きされます。ADC はオーバーラン条件を検出しません。
その他の DMA 対応周辺モジュール	データ オーバーランは発生しません。

DMA が周辺モジュールからメモリへデータを移動中である場合にのみ、ハードウェアでデータ オーバーランを検出できます。バッファエンプティ割り込み等に基づくメモリから周辺モジュールへの DMA データ転送は常に実行されます。結果として生じるメモリデータ オーバーランは、ソフトウェアを使って検出する必要があります。重複した DMA 要求は無視され、保留中の要求はそのまま保留されます。通常のように、DMA チャンネルは転送が完了した時に DMA 要求をクリアします。この際に CPU が介入しないと、その転送データが最新 (オーバーラン) データとなり、先のデータは失われます。

ユーザ アプリケーションは、データソースの性質に応じて複数の方法でオーバーラン エラーを処理できます。DMAC のデータリカバリとそのデータソース / シンクとの同期再確立は、アプリケーションに大きく依存するタスクです。ストリーミングデータ (DCI 周辺モジュール経由のコーデックからのデータ等) の場合、アプリケーションは失われたデータを無視できます。DMA 割り込みハンドラは、可能であれば問題の原因を修正した後に、データが再度正しくバッファされるように DMAC と DCI の同期再確立を試みます。ユーザ アプリケーションは、さらにオーバーランが発生する事を防ぐために、迅速に対応する必要があります。

周辺モジュール オーバーラン割り込みに移行するまでに、保留状態の DMA 要求は、失われたデータが移動されるはずであったアドレスにオーバーラン データ値の移動を完了しています。そのデータを正しいアドレスに移動し、欠損データスロットに NULL データ値を挿入する事ができます。その後、チャンネルのメモリアドレスを適切に調整できます。エラーのあったチャンネルに対する後続の DMA 要求は、修正済みメモリアドレスに対して通常通り転送を開始します。データ損失を許容できないアプリケーションの場合、周辺モジュール オーバーラン割り込みによってデータが失われる前に現在のブロック転送を中止し、DMA チャンネルを再初期化した後に、データの再送を要求する必要があります。

© 2012 Microchip Technology Inc.

22.9 デバッグのサポート

デバッグ中の DMA 動作の監視を容易にするために、DMA コントローラは各種のステータス レジスタを備えています。これらのレジスタは、直前に転送を実行した DMA チャンネル (DMALCA レジスタの LSTCH<3:0> ビット) と、そのチャンネルがアクセスしていたメモリアドレス オフセット (DSADR レジスタの DSADR<21:0>) および使っていたバッファ (DMACS1 レジスタの PPSTx ビット) に関する情報を提供します。

22.10 データ書き込みコリジョンと要求コリジョン

22.10.1 データ書き込みコリジョン

CPU と DMA チャンネルは、同時に任意の DPSRAM または DMA 対応周辺モジュール データ レジスタを読み出すか、あるいは一方が読み出し中に他方が書き込めます。唯一の制約条件として、CPU と DMA チャンネルの両方が同時に同一アドレスに書き込む事はできません。通常、このような状況は発生しません。しかし何らかの理由によりこのような状況が発生した場合、CPU 書き込みを優先させる事もできますが、これは主に予測可能な動作を行うためであり、それ以外にほとんど実用性はありません。

同一バスサイクル中に同一アドレスに対して DMA チャンネルによる書き込みと CPU による読み出し (あるいは CPU による書き込みと DMA チャンネルによる読み出し) を実行できます。しかしこの場合、そのバスサイクル中に書き込まれた値ではなく、以前の値が読み出されます。また、このような状況は正常動作とみなされ、特別な対応動作は発生しません。

CPU と DMA チャンネルが同時に同一周辺モジュール アドレスに書き込みアクセスした場合、DMA 周辺モジュール書き込みコリジョン ステータス レジスタ (DMAPWC) の PWCOLx ビットがセットされます。全てのコリジョン ステータスフラグ値の論理演算により、共通の DMAC フォルトトラップが生成されます。PWCOLx フラグは、ユーザ アプリケーションが割り込みコントローラ レジスタ (INTCON1) の DMAC エラー ステータスビット (DMACERR) をクリアした時に、自動的にクリアされます。

PWCOLx がクリアされるまで、書き込みコリジョン エラーが発生したチャンネルに対する後続の DMA 要求は無視されます。

表 22-4 に、各種の同時発生イベントに対するデバイスの対応を要約して示します。

表 22-4: デュアルポート エレメントの同時アクセス規則

アクセス タイプ	CPU 動作	DMA 読み出し	DMA 書き込み	備考
DPSRAM 同時 アクセス	CPU 読み出し	OK	—	CPU と DMA の同時読み出しは許容される
	CPU 読み出し	—	OK	DMA は CPU が読み出したデータを上書きする
	CPU 書き込み	OK	—	CPU は DMA が読み出したデータを上書きする
	CPU 書き込み	—	CPU を優先	DMA 書き込みは無視される
SFR 同時 アクセス	CPU 読み出し	OK	—	CPU と DMA の同時読み出しは許容される
	CPU 読み出し	—	OK	DMA は CPU が読み出したデータを上書きする
	CPU 書き込み	OK	—	CPU は DMA が読み出したデータを上書きする
	CPU 書き込み	—	CPU を優先	DMA 書き込みは無視され、PWCOLx フラグがセットされる

例 22-13 に、DMA チャンネル 0 で DPSRAM から周辺モジュール (UART) にデータを転送し、DMA チャンネル 1 で周辺モジュール (ADC) から DPSRAM にデータを転送する場合の DMA コントローラのトラップ処理用サンプルコードを示します。

例 22-13:DMA コントローラのトラップ処理

```
void __attribute__((__interrupt__,no_auto_psv)) _DMACError(void)
{
 static unsigned int ErrorLocation;

 // Peripheral Write Collision Error Location
 if(DMAPWC & 0x0001)
 {
    ErrorLocation = DMA0STA;
 }

 // DMA RAM Write Collision Error Location
 if(DMARQC & 0x0002)
 {
   ErrorLocation = DMA1STA;
 }

 INTCON1bits.DMACERR = 0;         //Clear Trap Flag
}
```

22.10.2 チャンネル要求の衝突

割り込みによる DMA 要求が処理中、保留もしくは同一チャンネルへの手動で転送を開始 (FORCE = 1) が同時に起こった場合、一般的に手動での要求は通常に破棄 (無視) され割り込みによる要求が優先されます。要求の衝突が検出されると、DMA 要求コリジョン レジスタ (DMARQC) 内の対応するフラグ (RQCOLx) がセットされます。

DMA 転送はマルチサイクル のイベントであるため、ユーザ手動や割り込みによる転送要求はお互いに異なるタイミングで発生し、その結果要求の衝突が起こります。つまり、衝突とは、手動転送要求と割り込みによる転送要求の重複が発生したサイクルとして定義されます。

要求が衝突した場合は下記のように処理されます。

1. 割り込みによる要求と手動要求 (FORCE ビットのセット) が同時に発生した衝突では、手動要求が破棄されて FORCE ビットが自動的にクリアされ、対応する RQCOL ビットがセットされます。割り込みによる要求は処理されます。
2. 割り込みによる要求によって転送が始まった後に手動要求 (FORCE ビットのセット) が発生した衝突では、手動要求が破棄されて FORCE ビットが自動的にクリアされ、対応する RQCOL ビットがセットされます。割り込みによる要求は処理されます。
3. 手動要求 (FORCE ビットのセット) によって転送が始まった後に割り込みによる転送要求が届いた衝突では、手動要求による転送が続行され、割り込みによる要求はその転送が完了するまで保留されます。転送が完了すると FORCE ビットは自動的にクリアされ、対応する RQCOL ビットがセットされます。その後に、割り込みによる要求が処理されます。

© 2012 Microchip Technology Inc.

22.11 省電力モード時の動作

22.11.1 スリープモード

スリープモード時に DMA は無効になります。スリープモードに移行する前に、全ての DMA チャンネルの未完了ブロック転送を完了させるか、あるいは無効にする事を推奨します。ただしこれは必須の措置ではありません。

22.11.2 アイドルモード

DMA はシステム内の第 2 のバスマスタであるため、CPU がアイドルモードに移行しても、データ転送を続ける事ができます。周辺モジュールが DMA チャンネルを使用し、かつその周辺モジュールがアイドルモード時にも動作するように設定されている場合、その周辺モジュールとメモリ間で双方向にデータを転送できます。ブロック転送完了時に、DMA チャンネルは割り込みを生成して CPU を復帰させます (その割り込みが有効になっている場合のみ)。復帰した CPU は割り込みサービスハンドラを実行します。

各周辺モジュールは、アイドル時停止制御ビットを備えています。この制御ビットをセットすると、その周辺モジュールは CPU がアイドルモードである間無効になります。周辺モジュールとのデータ転送 (方向問わず) に DMA を使う場合、周辺モジュールのアイドル時停止機能を有効にすると、実質的にその周辺モジュールに割り当てられている DMA チャンネルも無効になります。

22.12 レジスタマップ

dsPIC33E/PIC24E のダイレクト メモリアクセス (DMA) モジュールに関連するレジスタの要約を表 22-5 に示します。

表 22-5: DMA 関連のレジスタマップ

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
DMAxCON	CHEN	SIZE	DIR	HALF	NULLW	—	—	—	—	—	AMODE<1:0>		—	—	MODE<1:0>		0000
DMAxREQ	FORCE	—	—	—	—	—	—	—	IRQSEL<7:0>								0000
DMAxSTAH	—	—	—	—	—	—	—	—	STA<23:16>								0000
DMAxSTAL	STA<15:0>																0000
DMAxSTBH	—	—	—	—	—	—	—	—	STB<23:16>								0000
DMAxSTBL	STB<15:0>																0000
DMAxPAD	PAD<15:0>																0000
DMAxCNT	CNT<15:0>																0000
DSADR	DSADR<15:0>																0000
DMAPWC(1)	—	PWCOL14	PWCOL13	PWCOL12	PWCOL11	PWCOL10	PWCOL9	PWCOL8	PWCOL7	PWCOL6	PWCOL5	PWCOL4	PWCOL3	PWCOL2	PWCOL1	PWCOL0	0000
DMARQC(1)	—	RQCOL14	RQCOL13	RQCOL12	RQCOL11	RQCOL10	RQCOL9	RQCOL8	RQCOL7	RQCOL6	RQCOL5	RQCOL4	RQCOL3	RQCOL2	RQCOL1	RQCOL0	0000
DMALCA(1)	—	—	—	—	—	—	—	—	—	—	—	—	LSTCH<3:0>				000F
DMAPPS(1)	—	PPST14	PPST13	PPST12	PPST11	PPST10	PPST9	PPST8	PPST7	PPST6	PPST5	PPST4	PPST3	PPST2	PPST1	PPST0	0000

凡例 : — = 未実装、「0」として読み出し、リセット値は 16 進数で表記

Note 1: DMA チャンネルの数は製品によって異なります。詳細は**「ダイレクトメモリ アクセス (DMA)」**を参照してください。

© 2012 Microchip Technology Inc.

22.13 関連アプリケーション ノート

ダイレクトメモリ アクセスの使用に関連するアプリケーション ノートの一覧を以下に記載します。一部のアプリケーション ノートは dsPIC33E/PIC24E デバイスファミリ向けではありませんが、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。ダイレクト メモリアクセス (DMA) モジュールに関連する最新のアプリケーション ノートは下記の通りです。

タイトル	アプリケーション ノート番号
現在、関連するアプリケーション ノートはありません。	N/A

Note: dsPIC33E/PIC24E デバイス ファミリ関連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。

22.14 改訂履歴

リビジョン A (2009 年 4 月)

本書の初版

リビジョン B (2011 年 5 月)

このリビジョンでの変更内容は以下の通りです。

- 例 :
  - 例 22-2 ～例 22-13 のコードを更新
- 図 :
  - 図 22-4 ～図 22-6 を更新
  - 図 22-6 の表題を更新
  - 図 22-7 を追加
- Note:
  - 図 22-6 に Note を追加
- レジスタ :
  - レジスタ 22-8 の bit 15 と bit 14 の特性を「R/W-0」から「U-0」に変更
  - レジスタ 22-11 の RQCOLx ビットの名称を「チャンネル x 周辺モジュール書き込みコリジョン フラグビット」から「チャンネル x 転送要求コリジョン フラグビット」に変更
- その他の変更内容 :
  - **22.5.3「メモリアドレスの初期化」**を更新
  - **22.6.10「手動転送モード」**で、FORCE ビットをセットするタイミングに応じて FORCE ビットに適用可能な特殊な条件を更新
- 表 :
  - 表 22-2 内の UART の「設定における留意事項」を更新
- 「SMPI<3:0>」を全て「SMPI<4:0>」に更新
- マニュアルの名称を「dsPIC33E」から「dsPIC33E/PIC24E」ファミリ リファレンス マニュアルに更新
- 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 C (2011 年 12 月)

このリビジョンでの変更内容は以下の通りです。

- **22.1「はじめに」**内の第 2 段落を更新
- DMA コントローラのブロック図 (図 22-1 参照) に Note 1 を追加
- DMAxREQ: DMA チャンネル xIRQ 選択レジスタ (レジスタ 22-2) の IRQSEL<7:0> ビットの定義を更新し、Note 2 を削除
- 下記のレジスタの説明に Note 1 を追加 :
  - DMAPWC: DMA 周辺モジュール書き込みコリジョン ステータス レジスタ (レジスタ 22-10)
  - DMARQC: DMA 要求コリジョン ステータス レジスタ (レジスタ 22-11)
  - DMALCA: DMA 直前アクティブ チャンネル ステータス レジスタ (レジスタ 22-12)
  - DMAPPS: DMA ピンポン ステータス レジスタ (レジスタ 22-13)
- **22.5.3「メモリアドレスの初期化」**の第 1 段落を更新
- 54 Kbyte RAM を備えた dsPIC33E/PIC24E デバイスのデータメモリマップ (図 22-4) を更新
- DMA コントローラ トラップ処理のサンプルコード (例 22-13) を更新
- DMA 関連のレジスタマップ (表 22-5) の DMAPWC、DMARQC、DMALCA、DMAPPS レジスタに関する Note 1 と参照先を追加

© 2012 Microchip Technology Inc.

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性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、マイ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。マイ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

本書に記載されているデバイス アプリケーション等に関する情報は、ユーザの便宜のためにのみ提供されているものであり、更新によって無効とされる事があります。お客様のアプリケーションが仕様を満たす事を保証する責任は、お客様にあります。マイクロチップ社は、明示的、暗黙的、書面、口頭、法定のいずれであるかを問わず、本書に記載されている情報に関して、状態、品質、性能、商品性、特定目的への適合性をはじめとする、いかなる類の表明も保証も行いません。マイクロチップ社は、本書の情報およびその使用に起因する一切の責任を否認します。マイクロチップ社の明示的な書面による承認なしに、生命維持装置あるいは生命安全用途にマイクロチップ社の製品を使用する事は全て購入者のリスクとし、また購入者はこれによって発生したあらゆる損害、クレーム、訴訟、費用に関して、マイクロチップ社は擁護され、免責され、損害をうけない事に同意するものとします。暗黙的あるいは明示的を問わず、マイクロチップ社が知的財産権を保有しているライセンスは一切譲渡されません。

商標

マイクロチップ社の名称とMicrochipロゴ、dsPIC、FlashFlex、KEELOQ、KEELOQ ロゴ、MPLAB、PIC、PICmicro、PICSTART、PIC32 ロゴ、rfPIC、SST、SST ロゴ、SuperFlash、UNI/O は、米国およびその他の国におけるマイクロチップ・テクノロジー社の登録商標です。

FilterLab、Hampshire、HI-TECH C、Linear Active Thermistor、MTP、SEEVAL、Embedded Control Solutions Company は、米国におけるマイクロチップ・テクノロジー社の登録商標です。

Silicon Storage Technology は、その他の国におけるマイクロチップ・テクノロジー社の登録商標です。

Analog-for-the-Digital Age、Application Maestro、BodyCom、chipKIT、chipKIT ロゴ、CodeGuard、dsPICDEM、dsPICDEM.net、dsPICworks、dsSPEAK、ECAN、ECONOMONITOR、FanSense、HI-TIDE、In-Circuit Serial Programming、ICSP、Mindi、MiWi、MPASM、MPF、MPLAB 認証ロゴ、MPLIB、MPLINK、mTouch、Omniscient Code Generation、PICC、PICC-18、PICDEM、PICDEM.net、PICkit、PICtail、REAL ICE、rfLAB、Select Mode、SQI、Serial Quad I/O、Total Endurance、TSHARC、UniWinDriver、WiperLock、ZENA、Z-Scale は、米国およびその他の国におけるマイクロチップ・テクノロジー社の登録商標です。

SQTP は、米国におけるマイクロチップ・テクノロジー社のサービスマークです。

GestICとULPPは、その他の国におけるMicrochip Technology Germany II GmbH & Co. & KG (マイクロチップ・テクノロジー社の子会社) の登録商標です。

その他、本書に記載されている商標は各社に帰属します。

©2012, Microchip Technology Incorporated, Printed in the U.S.A., All Rights Reserved.

ISBN: 978-1-62076-556-2

QUALITY MANAGEMENT SYSTEM
CERTIFIED BY DNV
== ISO/TS 16949 ==

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-16949:2009 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://www.microchip.com/support
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

インディアナポリス
Noblesville, IN
Tel:317-773-8323
Fax:317-773-5453

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:86-10-8569-7000
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 - 杭州
Tel:86-571-2819-3187
Fax:86-571-2819-3189

中国 - 香港 SAR
Tel:852-2943-5100
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:86-755-8864-2200
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 - 大阪
Tel:81-6-6152-7160
Fax:81-6-6152-9310

日本 - 東京
Tel:81-3-6880- 3770
Fax:81-3-6880-3771

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 または
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:886-3-5778-366
Fax:886-3-5770-955

台湾 - 高雄
Tel:886-7-213-7828
Fax:886-7-330-9305

台湾 - 台北
Tel:886-2-2508-8600
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

11/29/12

©2012 Microchip Technology Inc.