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 20. データコンバータ インターフェイス (DCI)

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。

20.1 はじめに

データコンバータ インターフェイス (DCI) モジュールを使用すると、dsPIC33F とオーディオコーデック (コーダ / デコーダ) 等のオーディオ デバイス、AD コンバータ、DA コンバータを簡単に接続できます。

以下のインターフェイスをサポートしています。

- フレーム同期式シリアル転送 (シングルまたはマルチチャンネル)
- インター IC サウンド (I^2S) インターフェイス
- AC-LINK 準拠モード

音響アプリケーション用のコーデックの多くは 8 ～ 48 kHz のサンプリング レートをサポートしており、上記インターフェイス プロトコルのうちの 1 つを使用します。DCI はそれらのコーデックのインターフェイス タイミングを自動的に処理します。要求したデータ量を DCI モジュールで送受信するまで CPU のオーバーヘッドはかかりません。

DCI のデータワード長は、音響アプリケーションのデータサイズに合わせて最大 16 ビットまで設定可能です。しかし、多くのコーデックのデータワード サイズは 16 ビットを超えています。DCI はロングワードのデータ長をサポートできます。DCI は、複数の 16 ビット タイムスロットを使ってロングワードを送受信するように設定されています。この動作はユーザ アプリケーションに対して透過的です。ロングワード データは連続するレジスタ位置に格納します。

DCI は 1 つのデータフレームで最大 16 のタイムスロットをサポート可能で、最大 256 ビットのフレームサイズに対応しています。データフレームの各タイムスロットの制御ビットにより、タイムスロット中に DCI が送信するのか、受信するのかが決まります。

dsPIC33F DMA モジュールを使用すると、デュアルポート SRAM と DCI 送受信レジスタ間でデータの直接転送が可能です。

20.2 制御レジスタの説明

DCI は 5 つの制御レジスタと 1 つのステータス レジスタを備えます。

- **DCICON1: データ コンバータ インターフェイス モジュール制御レジスタ 1**
  DCI モジュール イネーブルビットとモードビットを制御します。
- **DCICON2: データ コンバータ インターフェイス モジュール制御レジスタ 2**
  DCI モジュールのワード長、データフレーム長、バッファ設定を制御します。
- **DCICON3: データ コンバータ インターフェイス モジュール制御レジスタ 3**
  DCI モジュールのビットクロック ジェネレータ設定を制御します。
- **DCISTAT: データ コンバータ インターフェイス モジュール ステータス レジスタ**
  DCI モジュールのステータス情報を提供します。
- **RSCON: 受信スロット イネーブル レジスタ**
  データを受信するためのアクティブ フレーム タイムスロット制御を有効にします。
- **TSCON: 送信スロット イネーブル レジスタ**
  データを送信するためのアクティブ フレーム タイムスロット制御を有効にします。

これらの制御レジスタとステータス レジスタに加えて、TXBUF0 から TXBUF3 の 4 つの送信レジスタと、RXBUF0 から RXBUF3 の 4 つの受信レジスタがあります。

レジスタ 20-1: DCICON1: データ コンバータ インターフェイス モジュール制御レジスタ 1

R/W-0	U-0	R/W-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0
DCIEN	—	DCISIDL	—	DLOOP	CSCKD	CSCKE	COFSD
bit 15							bit 8

R/W-0	R/W-0	R/W-0	U-0	U-0	U-0	R/W-0	R/W-0
UNFM	CSDOM	DJST	—	—	—	COFSM<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」= ビットをセット	「0」= ビットをクリア　　x = ビットは未知

bit 15 **DCIEN:** DCI モジュール イネーブルビット
1 = モジュールを有効にする
0 = モジュールを無効にする

bit 14 **予約 :** 「0」として読み出し

bit 13 **DCISIDL:** DCI アイドル時停止制御ビット
1 = モジュールは CPU アイドルモード中は動作を停止する
0 = モジュールは CPU アイドルモード中も動作を継続する

bit 12 **予約 :** 「0」として読み出し

bit 11 **DLOOP:** デジタル ループバック モード制御ビット
1 = デジタル ループバック モードを有効にし、CSDI ピンと CSDO ピンを内部で接続する
0 = デジタル ループバック モードを無効にする

bit 10 **CSCKD:** サンプルクロック方向制御ビット
1 = DCI モジュール有効時に CSCK ピンを入力とする
0 = DCI モジュール有効時に CSCK ピンを出力とする

bit 9 **CSCKE:** サンプルクロック エッジ制御ビット
1 = データはシリアルクロックの立ち下がりエッジで変化し、シリアルクロックの立ち上がりエッジでサンプリングする
0 = データはシリアルクロックの立ち上がりエッジで変化し、シリアルクロックの立ち下がりエッジでサンプリングする

bit 8 **COFSD:** フレーム同期方向制御ビット
1 = DCI モジュール有効時に COFS ピンを入力にする
0 = DCI モジュール有効時に COFS ピンを出力にする

bit 7 **UNFM:** アンダーフロー モードビット
1 = 送信アンダーフロー時は送信レジスタに最後に書き込まれた値を送信する
0 = 送信アンダーフロー時は「0」を送信する

bit 6 **CSDOM:** シリアルデータ出力モードビット
1 = 無効な送信タイムスロットの間、CSDO ピンを 3 ステートにする
0 = 無効な送信タイムスロットの間、CSDO ピンを「0」に駆動する

bit 5 **DJST:** DCI データ位置調整制御ビット
1 = データ送受信をフレーム同期パルスと同じシリアルクロック サイクル内に開始する
0 = データ送受信をフレーム同期パルスより 1 シリアルクロック サイクル後に開始する

bit 4-2 **予約 :** 「0」として読み出し

bit 1-0 **COFSM<1:0>:** フレーム同期モードビット
11 = 20 ビット AC-LINK モード
10 = 16 ビット AC-LINK モード
01 = I^2S フレーム同期モード
00 = マルチチャンネル フレーム同期モード

レジスタ 20-2: DCICON2: データ コンバータ インターフェイス モジュール制御レジスタ 2

U-0	U-0	U-0	U-0	R/W-0	R/W-0	U-0	R/W-0
—	—	—	—	BLEN<1:0>		—	COFSG3
bit 15							bit 8

R/W-0	R/W-0	R/W-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0
COFSG<2:0>			—	WS<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」= ビットをセット	「0」= ビットをクリア	x = ビットは未知

bit 15-12 **予約 :**「0」として読み出し

bit 11-10 **BLEN<1:0>:** バッファ長制御ビット
11 = 割り込みの間に 4 データワードをバッファする
10 = 割り込みの間に 3 データワードをバッファする
01 = 割り込みの間に 2 データワードをバッファする
00 = 割り込みの間に 1 データワードをバッファする

bit 9 **予約 :**「0」として読み出し

bit 8-5 **COFSG<3:0>:** フレーム同期ジェネレータ制御ビット
1111 = 16 ワードのデータフレーム
•
•
•
0010 = 3 ワードのデータフレーム
0001 = 2 ワードのデータフレーム
0000 = 1 ワードのデータフレーム

bit 4 **予約 :**「0」として読み出し

bit 3-0 **WS<3:0>:** DCI データワード サイズ ビット
1111 = データワード サイズは 16 ビット
•
•
•
0100 = データワード サイズは 5 ビット
0011 = データワード サイズは 4 ビット
0010 = 無効 (予測できない結果が生じる可能性があるため使用不可)
0001 = 無効 (予測できない結果が生じる可能性があるため使用不可)
0000 = 無効 (予測できない結果が生じる可能性があるため使用不可)

レジスタ 20-3: DCICON3: データ コンバータ インターフェイス モジュール制御レジスタ 3

U-0	U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	—	—	BCG<11:8>			
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
BCG<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」= ビットをセット	「0」= ビットをクリア	x = ビットは未知

bit 15-12 **予約 :** 「0」として読み出し

bit 11-0 **BCG<11:0>:** DCI ビットクロック ジェネレータ制御ビット

レジスタ 20-4: DCISTAT: データ コンバータ インターフェイス モジュール ステータス レジスタ

U-0	U-0	U-0	U-0	R-0	R-0	R-0	R-0
—	—	—	—	SLOT<3:0>			
bit 15							bit 8

U-0	U-0	U-0	U-0	R-0	R-0	R-0	R-0
—	—	—	—	ROV	RFUL	TUNF	TMPTY
bit 7							bit 0

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

bit 15-12 **予約** : 「0」として読み出し

bit 11-8 **SLOT<3:0>:** DCI スロット ステータスビット
1111 = スロット 15 は現在アクティブである
•
•
•
0010 = スロット 2 は現在アクティブである
0001 = スロット 1 は現在アクティブである
0000 = スロット 0 は現在アクティブである

bit 7-4 **予約** : 「0」として読み出し

bit 3 **ROV:** 受信オーバーフロー ステータスビット
1 = 少なくとも 1 つの受信レジスタでオーバーフローが発生した
0 = 受信オーバーフローは発生していない

bit 2 **RFUL:** 受信バッファフル ステータスビット
1 = 受信レジスタに新しいデータがある
0 = 受信レジスタに以前のデータがある

bit 1 **TUNF:** 送信バッファ アンダーフロー ステータスビット
1 = 少なくとも 1 つの送信レジスタで送信アンダーフローが発生した
0 = 送信アンダーフローは発生していない

bit 0 **TMPTY:** 送信バッファ エンプティ ステータスビット
1 = 送信レジスタはエンプティである
0 = 送信レジスタはエンプティではない

レジスタ 20-5: RSCON: 受信スロット イネーブル 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
RSE15	RSE14	RSE13	RSE12	RSE11	RSE10	RSE9	RSE8
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
RSE7	RSE6	RSE5	RSE4	RSE3	RSE2	RSE1	RSE0
bit 7							bit 0

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

bit 15-0 **RSE<15:0>:** 受信スロット イネーブルビット
1 = タイムスロット 15 で CSDI データを受信する
0 = タイムスロット 15 で CSDI データを無視する
•
•
•
1 = タイムスロット 0 で CSDI データを受信する
0 = タイムスロット 0 で CSDI データを無視する

レジスタ 20-6: TSCON: 送信スロット イネーブル 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
TSE15	TSE14	TSE13	TSE12	TSE11	TSE10	TSE9	TSE8
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
TSE7	TSE6	TSE5	TSE4	TSE3	TSE2	TSE1	TSE0
bit 7							bit 0

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

bit 15-0 **TSE<15:0>:** 送信スロット イネーブル制御ビット
1 = 送信バッファの内容をタイムスロット 15 で送信する
0 = CSDO ピンは、タイムスロット 15 で CSDOM ビットの状態に応じて、3 ステートにするか「0」に駆動する
•
•
•
1 = 送信バッファの内容をタイムスロット 0 で送信する
0 = CSDO ピンは、タイムスロット 0 で CSDOM ビットの状態に応じて、3 ステートにするか「0」に駆動する

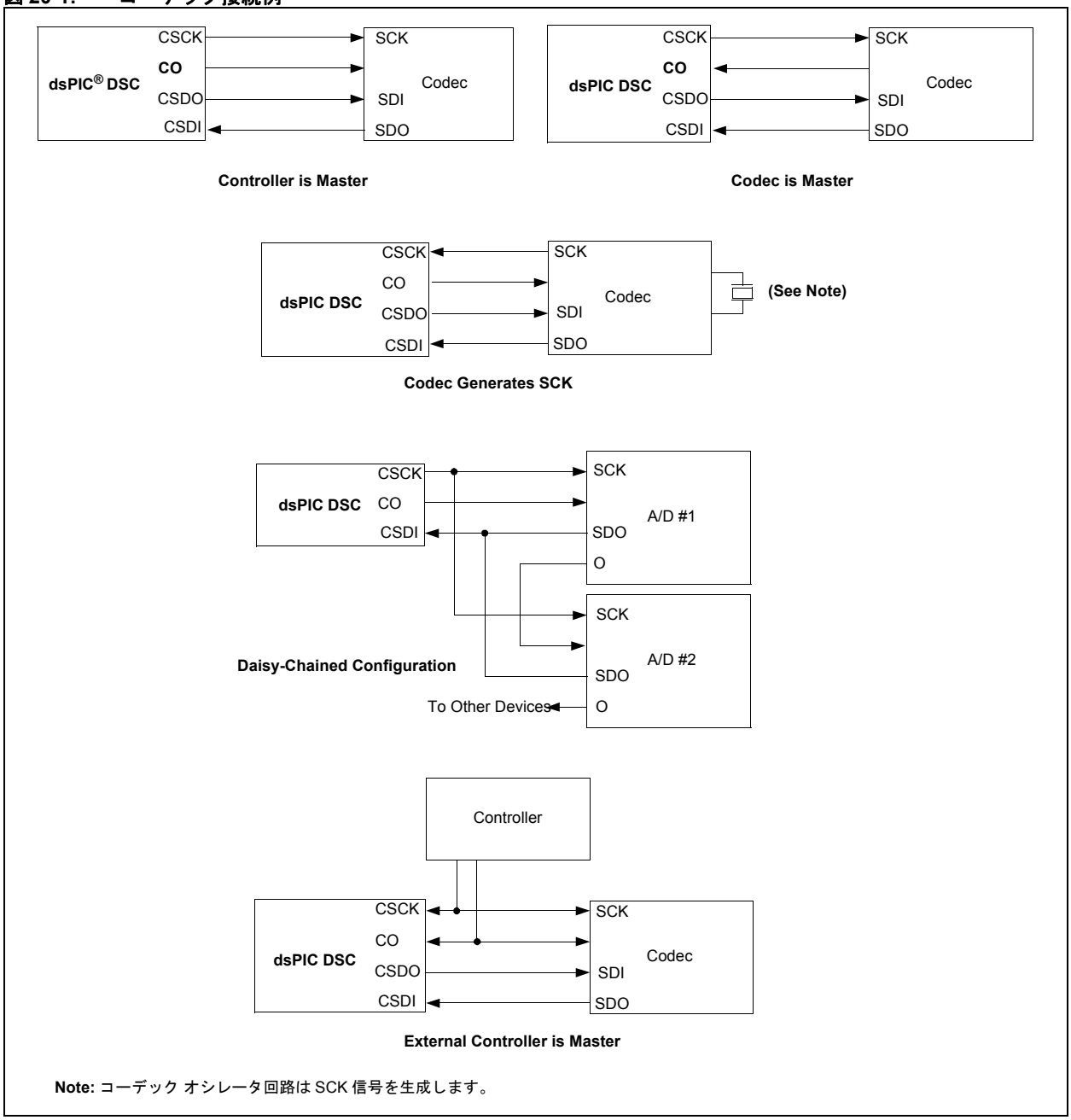
20.3 コーデック インターフェイスの基礎と用語

コーデック アプリケーションの最小単位はコントローラとコーデック デバイスです。DCI がサポートするインターフェイス プロトコルでは、2 つのデバイス間のデータ転送を開始するためにフレーム同期 (FS) 信号 (dsPIC33F では COFS) を使用する必要があります。通常、FS の立ち上がりエッジでデータの転送を開始します。いずれのデバイスも FS を生成できます。FS を生成するデバイスがマスタとなります。概念上、送受信デバイスどちらもマスタになれます。

図 20-1 に接続例を示します。FS 信号の周波数は通常システムのサンプリング レート (fs) です。

Note: 本セクションで説明する詳細は DCI モジュールだけに限定したものではありません。ここでは、ほとんどのコーデック デバイスで使用されているデジタル シリアル インターフェイス プロトコルに関連する背景と専門用語の一部を取り上げます。

図 20-1: コーデック接続例



Note: コーデック オシレータ回路は SCK 信号を生成します。

20.3.1 シリアル転送クロック

全てのインターフェイスはシリアル転送クロック SCK (dsPIC33F では CSCK ピン) を備えます。SCK 信号は接続されている全てのデバイスから生成されます。外部から供給する事も可能です。SCK をビットクロックと呼ぶシステムもあります。高信号忠実度を提供するコーデックの場合、コーデック デバイスの水晶振動子から SCK 信号を得るのが一般的です。プロトコルはサンプリングするデータの SCK エッジを定義します。マスタデバイスは SCK を基準にして FS 信号を生成します。

FS 信号の周期がデータフレームの長さを規定します。この周期はデータ サンプリング周期と同じです。データフレーム中に発生する SCK サイクルの数は、選択したコーデックタイプで決まります。システム サンプリング レートに対する SCK 周波数の比率は、n 比で表します。n はデータフレームあたりの SCK 周期の数を意味します。

20.3.2 データ転送とタイムスロット

データはシリアルデータ出力 (SDO) とシリアルデータ入力 (SDI) 信号 (dsPIC33F の CSDO ピンと CSDI ピン) を介して転送されます。フレーム同期式のインターフェイス プロトコルを使用する利点の 1 つは、サンプリング周期またはデータフレームごとに複数のデータワードを転送できる事です。例えば、入力チャンネルを 4 つ備える 16 ビットコーデックを想定してみましょう。コーデックは 1 つの FS 周期で 4 つの 16 ビットワードを送信する必要があります。このため、FS 周期ごとに 64 SCK サイクル、つまり n = 64 となります。

タイムスロットは複数のコーデック データチャンネルまたは制御情報に使用できます。さらに複数のデバイスを同じシリアル データピンに多重化できます。各スレーブデバイスは、適切なタイムスロット中にシリアルデータ接続上にデータを出力するようプログラムされます。それ以外のタイムスロットは、各スレーブデバイスの出力が他のデバイスがシリアルバスを使用できるように 3 ステートです。

フレーム同期出力 (FSO) ピンによって FS 信号をデイジーチェーン接続できるデバイスもあります。図 20-1 に、代表的なデイジーチェーン接続を示します。最初のスレーブデバイスからの転送が完了すると、FSO ピンを経由してチェーン内の 2 番目のデバイスに FS パルスが送られます。このプロセスは、チェーン内の最後のデバイスがデータを送信するまで続きます。コントローラ (マスタ) デバイスには、転送する最大サイズのデータワードが収まるデータ フレームサイズをプログラムする必要があります。

20.3.2.1 データ転送のタイミング

図 20-2 に、代表的なデータ転送タイミングを示します。ほとんどのプロトコルは、FS 信号を検出後 1 SCK サイクル経過してから転送を開始します。この例では、16 fs クロック (fs はサンプリング周波数) を使用してフレームあたり 4 つの 4 ビット データワードを転送しています。

図 20-2: フレーム同期式データ転送の例

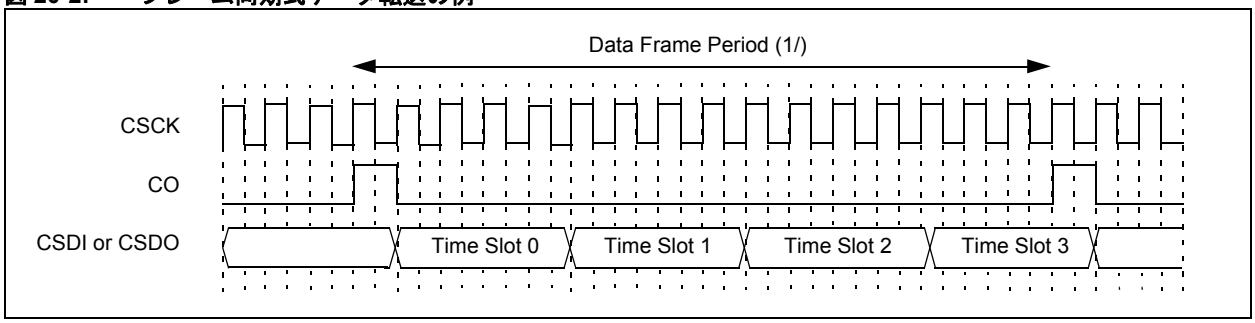
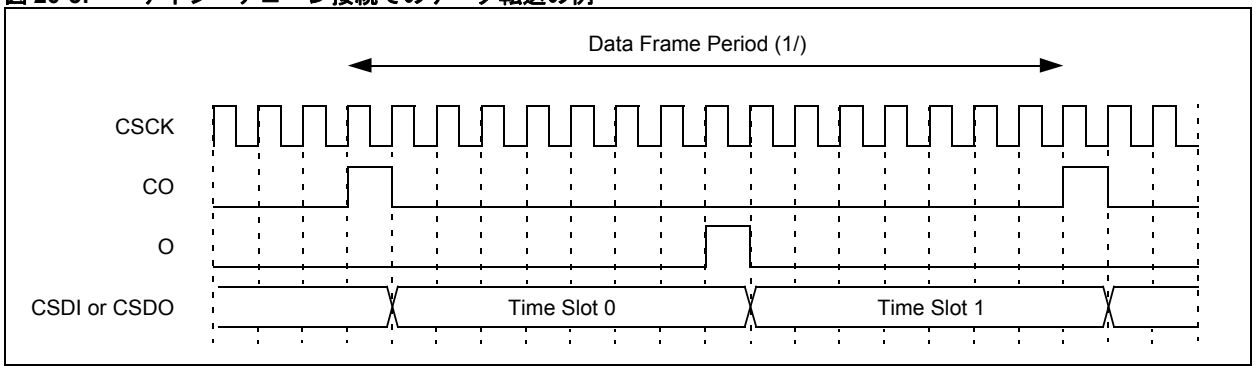


図 20-3 に、デバイスをデイジーチェーン接続した場合の代表的なデータ転送タイミングを示します。この例では、16 fs SCK 周波数を使用してフレームあたり 2 つの 8 ビット データワードを転送しています。FS パルスの検出後、チェーン内の 1 番目のデバイスは最初の 8 ビット データワードを転送し、転送完了時に FSO 信号を生成します。FSO 信号はチェーン内の 2 番目のデバイスから 2 番目のデータワードの転送を開始します。

図 20-3: デイジーチェーン接続でのデータ転送の例



20.3.3 FS パルス

スレーブデバイスがデータフレームの開始を検出できるよう、FS パルスには 1 回の SCK 周期における最小アクティブ時間があります。FS パルスのデューティ サイクルは、データフレームで境界を区切るプロトコルによって異なります。

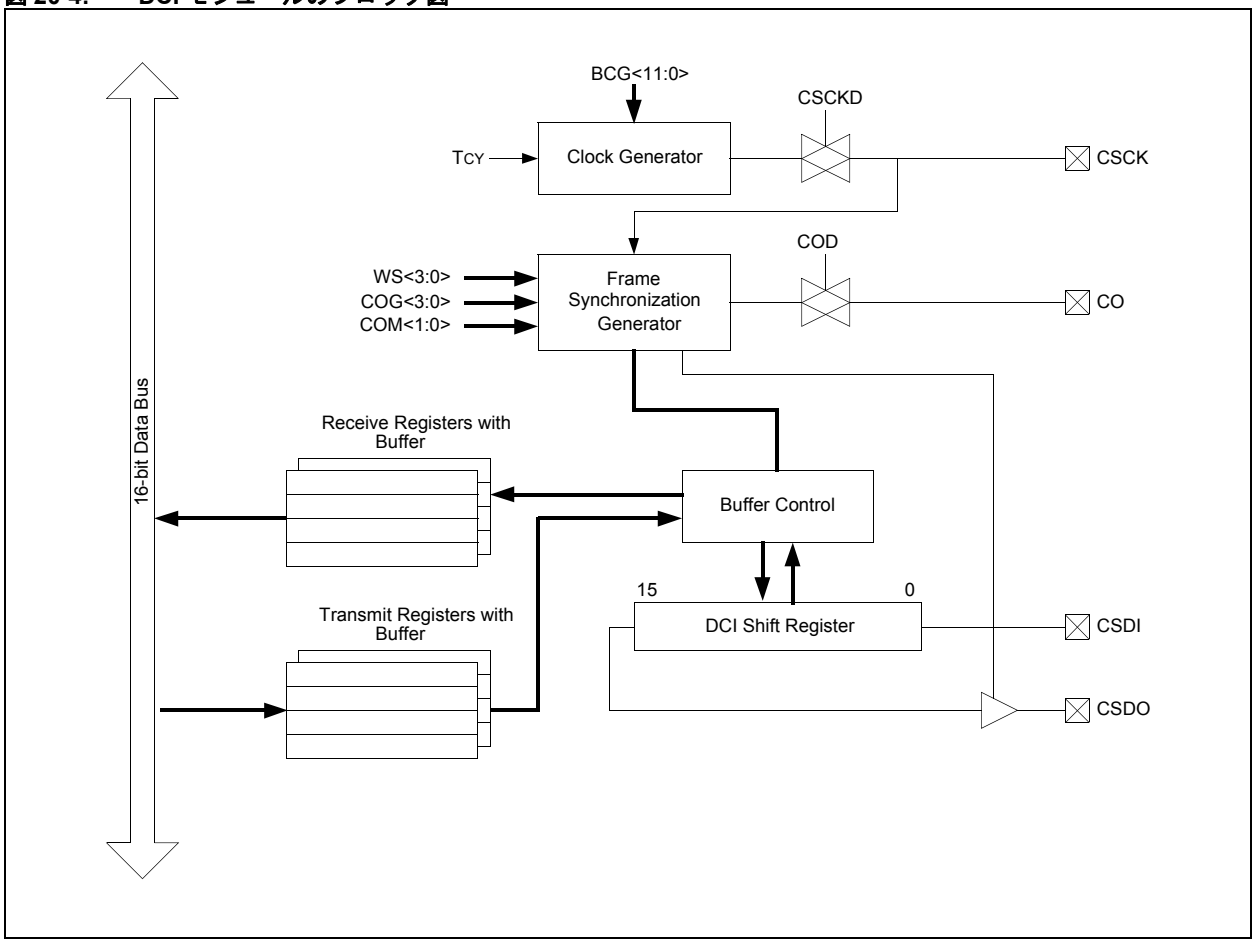
例えば、I^2S プロトコルは 50% のデューティ サイクルを持つ FS 信号を使用します。I^2S プロトコルは、2 つのデータチャンネル (左右チャンネル音声情報) の転送に最適化されています。FS 信号のエッジは左右のチャンネル データワードの境界を示します。

別の例として、AC-LINK プロトコルは 16 SCK 周期の間は High、240 SCK 周期の間は Low を保つ FS 信号を使用します。AC-LINK の FS 信号のエッジは、フレーム内の制御情報とデータの境界を示します。

20.4 DCI 動作

図 20-4 に DCI モジュールの概略ブロック図を示します。本モジュールは、送受信シフトレジスタがバッファ制御ユニットを経由して小規模なメモリバッファに接続される構成です。これにより、DCI は各種コーデック シリアル プロトコルをサポートできます。DCI シフトレジスタは 16 ビット幅です。データは DCI 最上位ビット (MSb) から送受信されます。

図 20-4: DCI モジュールのブロック図



20.4.1 DCI ピン

DCI モジュールに関連して 4 つの I/O ピン (CSCK、CSDO、CSDI、COFS) があります。DCI モジュールが有効の時、4 つのピンそれぞれのデータ方向を制御します。

20.4.1.1 CSCK ピン

CSCK ピンは、DCI にシリアルクロック接続を供給します。CSCK ピンは CSCKD 制御ビット (DCICON1<10>) を使用して、入力または出力に設定できます。

- CSCK ピンを出力に設定した場合 (CSCKD = 0)、dsPIC33F システムクロック源から得たシリアルクロックを DCI によって外部デバイスに供給します。
- CSCK ピンを入力に設定した場合 (CSCKD = 1)、外部デバイスからシリアルクロックを供給する必要があります。

20.4.1.2 CSDO ピン

DCI モジュール有効時、シリアルデータ出力 (CSDO) ピンは出力専用ピンに設定されます。CSDO ピンはデータ送信時にシリアル バスを駆動します。データを送信していないシリアルクロックの周期の間、CSDO ピンはシリアルデータ出力モード (CSDOM) 制御ビット (DCICON1<6>) の状態に応じて、3 ステートまたは「0」に駆動されます。3 ステートオプションを使うと、CSO 信号を他のデバイスが多重化して接続することができます。

20.4.1.3 CSDI ピン

DCI モジュール有効時、シリアルデータ入力 (CSDI) ピンは入力専用ピンに設定されます。

20.4.1.4 COFS ピン

フレーム同期 (COFS) ピンは、CSDO ピンと CSDI ピンで発生するデータ転送の同期をとるために使用します。COFS ピンは双方向性で、入力と出力のいずれかに設定可能です。COFS ピンのデータ方向は、COFSD 制御ビット (DCICON1<8>) で設定します。

- COFSD ビットがクリアされている場合、COFS ピンは出力です。DCI モジュールは FS パルスを生成してデータ転送を開始します。この設定では DCI がマスタデバイスです。
- COFSD ビットがセットされている場合、COFS ピンは入力です。DCI モジュールに同期信号が入力されるとデータ転送が開始します。COFSD 制御ビットがセットされている場合、DCI はスレーブデバイスです。

20.4.2 モジュールの有効化

DCI モジュールは、DCI モジュール イネーブル (DCIEN) 制御ビット (DCICON1<15>) をセット / クリアする事によって、有効化 / 無効化できます。DCIEN 制御ビットをクリアするとモジュールはリセットされます。シリアルクロックに関連するカウンタ、FS、バッファ制御ロジックは全てリセットされます。詳細は、**20.5.1.1「DCI 起動とデータ バッファリング」**と **20.5.1.2「DCI の無効化」**を参照してください。

DCI モジュールが有効の時、関連する CSCK、CSDI、CSDO、COFS I/O ピンのデータ方向を制御します。DCIEN ビット (DCICON1<15>) がセットされると、これらの I/O ピンの PORT、LAT、TRIS レジスタの値は DCI モジュールによってオーバーライドされます。

20.4.3 ビットクロック ジェネレータ

DCI モジュールにはビットクロックを生成する専用の 12 ビット タイムベースがあります。ビットクロック レート (周期) は、DCI ビットクロック ジェネレータ (BCG) 制御ビット (DCICON3<11:0>) に 0 以外の 12 ビット値を書き込んで設定します。BCG ビットが 0 に設定されている場合、ビットクロックは無効です。

Note: DCIEN ビットがセットされている場合、または DCICON3<11:0> に 0 以外の値を書き込んでビットクロック ジェネレータが有効になっている場合、CSCK I/O ピンは DCI モジュールに制御されています。こうすると、BCG は DCI モジュールから独立して動作できます。

CSCK ピンが DCI モジュールによって制御されている場合、CSCK ピンに対応する PORT、LAT、TRIS 制御レジスタの値はオーバーライドされ、CSCK ピンのデータ方向は CSCKD 制御ビット (DCICON1<10>) によって制御されます。

- DCI のシリアルクロックが外部デバイスから供給される場合、BCG ビット (DCICON3<11:0>) を「0」、CSCKD ビットを「1」に設定します。
- シリアルクロックが DCI モジュールによって生成される場合、BCG 制御ビット (DCICON3<11:0>) を 0 以外の値に設定し (式 20-1 参照)、CSCKD 制御ビット (DCICON1<10>) を「0」に設定します。

式 20-1 に、ビットクロック周波数を求める式を示します。

式 20-1: DCI ビットクロック ジェネレータの値

$$BCG<11:0> = \frac{FCY}{2\ FCSCK} - 1$$

必要なビットクロック周波数は、システム サンプリング レートとフレームサイズにより決まります。代表的なビットクロック周波数は、使用するデータコンバータと通信プロトコルによって異なりますが、コンバータ サンプリング レートの 16 ～ 512 倍です。

例えば、40 MIPS で動作している dsPIC33F を想定してみましょう。DCI モジュールは 16 ビット コーデックと接続する必要があり、8 kHz のサンプリングレートに設定されています。従って、FS 周期 = 1/8 kHz = 125 µs となります。

コーデックはフレームごとに 2 つの 16 ビットワードを送信し、サンプリング周波数でフレームを生成します。フレーム内の 2 つの 16 ビットワードのビット周期は、(125 µs/(2 x 16)) = 3.960625 µs である必要があります。従って、このコーデックのクロック周波数は FCSCK = (1/3.960625 µs) = 256 kHz です。

式 20-1 を当てはめると、DCI モジュールのビットクロック ジェネレータ (BCG) の値は BCG = [40000000/(2 x 256000)] - 1 = 77 です。

20.4.4 サンプルクロック エッジの選択

サンプルクロック エッジ (CSCKE) 制御ビット (DCICON1<9>) により、シリアルクロック信号のサンプリング エッジが決まります。

- CSCKE ビットがクリアされている場合 (既定値)、データは CSCK 信号の立ち下がりエッジでサンプリングされます。AC-LINK プロトコルとほとんどのマルチチャンネル フォーマットでは、データは CSCK 信号の立ち下がりエッジでサンプリングする必要があります。
- CSCKE ビットがセットされている場合、データは CSCK の立ち上がりエッジでサンプリングされます。I^2S プロトコルでは、データはシリアルクロック信号の立ち上がりエッジでサンプリングする必要があります。

20.4.5 フレーム同期モード制御ビット

DCI がサポートするインターフェイス プロトコルのタイプは、FS モード (COFSM) 制御ビット (DCICON1<1:0>) を使用して選択します。表 20-1 に、動作モード一覧を示します。

表 20-1: 動作モード

モード	DCICON1<1:0> の値	詳細記載セクション
マルチチャンネル	00	**20.5.4「マルチチャンネルの動作」**
I^2S	01	**20.5.5「I2S の動作」**
AC-LINK (16 ビット)	10	**20.5.6「AC-LINK の動作」**
AC-LINK (20 ビット)	11	

20.4.6 ワードサイズ選択ビット

DCI データワード サイズ (WS) ビット (DCICON2<3:0>) により、各 DCI データワードのビット数が決まります。これはフレーム内の各タイムスロットの長さです。データ長は 4 ～ 16 ビットの間で選択できます。16 ビットを超えるワードサイズは、複数のタイムスロットを使用する事で処理できます。詳細は、**20.5.3「ロングワード データ サポートのデータ パッキング」**を参照してください。

Note: WS 制御ビットはマルチチャンネル モードと I^2S モードでのみ使用します。これらのビットは、プロトコルでデータスロット サイズが固定されている AC-LINK モードには影響を与えません。

20.4.7 フレーム同期ジェネレータ

フレーム同期ジェネレータ (FSG) は、データワード内のフレーム長を設定する 4 ビット カウンタです。FSG の周期は、フレーム同期ジェネレータ (COFSG) 制御ビット (DCICON2<8:5>) で設定します。式 20-2 に、FSG 周期 (シリアルクロック サイクル数) を算出する式を示します。

式 20-2: フレーム長の CSCK サイクル数

$$FrameLength = (WS<3:0> + 1) \bullet (COG<3:0> + 1)$$

データフレームには、データが転送されないタイムスロットが含まれる事があります。例を挙げると、16 ビットのコーデックでは、16 ビットのデータワードを受信 (タイムスロット 0) した 16 クロックサイクル後 (タイムスロット 2) に制御ワードを受信する必要があります。このコーデックは、タイムスロット 0 (図 20-5 参照) で出力ラインにもデータワードを送信します。

フレームの全長は 3 ワード、すなわち 48 クロックサイクル (ワードあたり 16 クロックサイクル x 3 ワード) です。このコーデックで通信するには、DCICON レジスタビットを以下のように設定する必要があります。

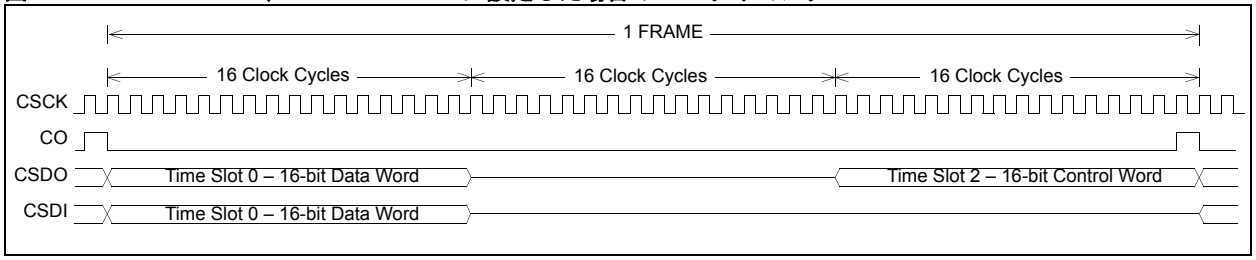
- ワードサイズ WS (DCICON2<3:0) = 1111 (16 ビット)
- フレーム同期ジェネレータ COFSG (DCICON2<8:5>) = 0010 (3 ワード)

タイムスロット 1 でデータが送信されていなくても、フレームは無効なタイムスロット (データが送受信されていないタイムスロット) を含む長さである必要があります。

フレーム長には最大で 16 データワードまでの値が選択できます。シリアルクロック周期のフレーム長は、選択するワードサイズに応じて異なります (最大で 256)。

Note: AC-LINK モードでは、プロトコルによってフレーム長が 256 シリアルクロック周期に設定されているため、COFSG 制御ビットの影響は受けません。

図 20-5: WS = 1111、COFSG = 0010 に設定した場合の DCI タイミング



20.4.8 レジスタの送受信

DCI には、TXBUF0 ～ 3 の 4 つの送信レジスタと、RXBUF0 ～ 3 の 4 つの受信レジスタがあります。送受信レジスタは全てメモリに割り当てられています。

20.4.8.1 バッファデータの配置

オーディオ PCM データは 2 の補数形式の符号付き小数で表される事から、データ値は TXBUF/RXBUF レジスタ内で常に左寄せで格納されます。プログラムされた DCI のワードサイズが 16 ビット未満の場合、受信レジスタの未使用の下位ビット (LSb) は「0」にセットされます。送信レジスタの未使用 LSb は無視されます。

20.4.8.2 バッファの送受信

送受信レジスタにはそれぞれ、ユーザ ソフトウェアからアクセスできないバッファがあります。送受信バッファ位置は、実質的にそれぞれダブルバッファ構造です。DCI は送信バッファからデータを送信し、受信バッファに受信データを書き込みます。バッファの働きにより、DCI がバッファからデータを使用している間に、ユーザ ソフトウェアは RXBUF レジスタと TXBUF レジスタを読み書きできます。

Note:	レジスタの TXBUF0 ～ 3 は書き込み専用レジスタです。ユーザは読み出しできません。

20.4.9 DCI バッファ制御ユニット

DCI モジュールは、バッファメモリとシリアル シフトレジスタ間でデータを転送するバッファ制御ユニットを搭載しています。バッファ制御ユニットは、バッファメモリと TXBUF/RXBUF レジスタ間でもデータを転送します。バッファ制御ユニットの働きにより、DCI は CPU にオーバーヘッドをかけずに送受信する複数のデータワードをキューイングできます。

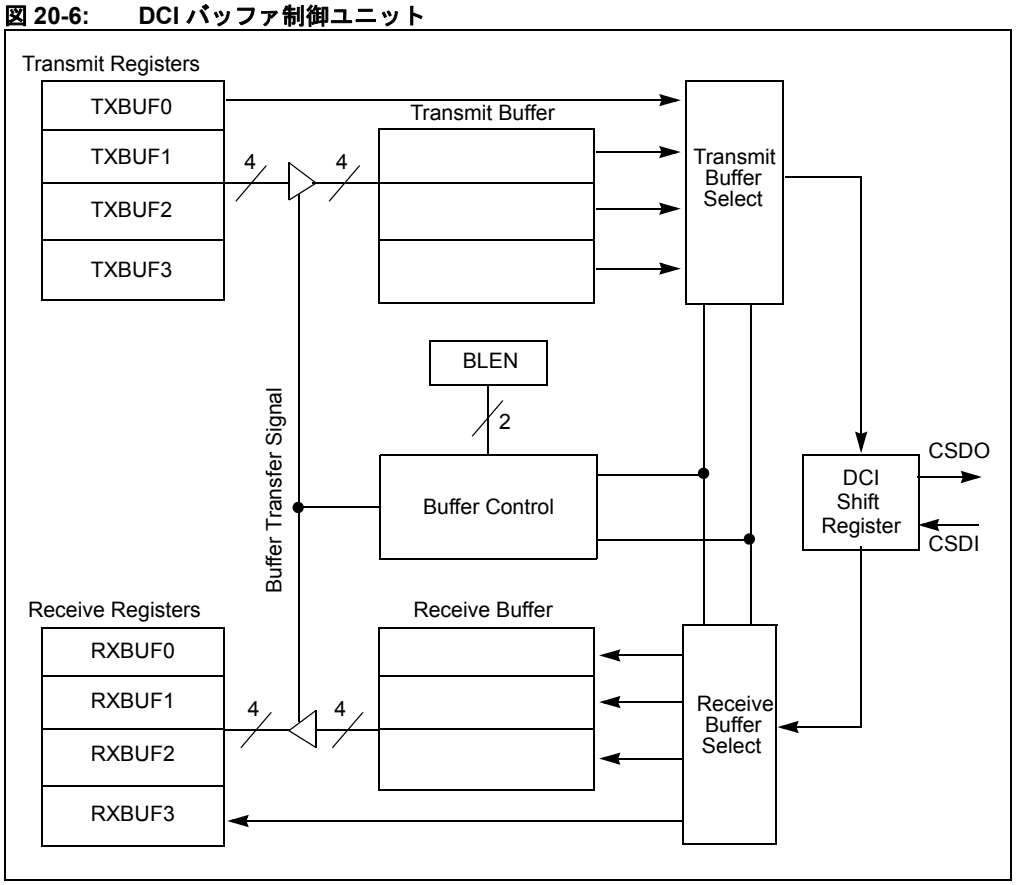
DCI は、バッファメモリと TXBUF/RXBUF レジスタ間の転送が実行されるたびに割り込みを発生させます。割り込みの間にバッファされるデータワード数は、バッファ長 (BLEN) 制御ビット (DCICON2<11:10>) によって決まります。送受信バッファのサイズは、BLEN 制御ビットの値で決めます (1 ～ 4 データワード)。

DCI シフトレジスタとバッファメモリ間でデータ転送が実行されるたびに、DCI バッファ制御ユニットは次のバッファ位置を指すようインクリメントされます。送受信データワード数が「BLEN 値 + 1」の場合、以下の動作が行われます。

1. バッファ制御ユニットは最初のバッファ位置を指すようリセットされる。
2. 受信バッファに格納されている受信データは RXBUF レジスタに転送される。
3. TXBUF レジスタにあるデータはバッファに転送される。
4. CPU 割り込みが発生する。

DCI バッファ制御ユニットは、常に送受信バッファの同じ相対位置にアクセスします。例えば DCI が TXBUF3 からデータを送信している場合、このタイムスロットの間に受信するデータは全て RXBUF3 に書き込まれます。

図 20-6: DCI バッファ制御ユニット



20.4.10 送信スロット イネーブルビット

送信スロット イネーブル (TSCON) レジスタには、最大で 16 の送信用タイムスロットを有効化できる送信スロット イネーブル (TSE) 制御ビット (TSCON<15:0>) があります。各タイムスロットのサイズは、DCI データワード サイズ (WS) ビット (DCICON2<3:0>) によって決まります。最大値は 16 ビットです。

TSE ビットのいずれかによって送信タイムスロットを有効にする場合 (TSEx = 1)、現在の送信バッファ位置の内容が CSDO シフトレジスタに書き込まれ、DCI バッファ制御ユニットは次のバッファ位置を指すようインクリメントされます。データを送信するには、少なくとも 1 つの送信タイムスロットが有効である必要があります。無効なタイムスロットが出現した場合、対応する TXBUFx レジスタの内容は送信されずにバッファポインタがインクリメントされます。

選択するフレームサイズが 16 データスロット未満の場合、全ての TSE 制御ビットがモジュールの動作に影響を与えるわけではありません。TSE 制御ビットの上位何ビットかは使用されません。例えば、COFSG<3:0> = 0111 (フレームあたり 8 データスロット) の場合、TSE8 ～ 15 は DCI の動作に影響を与えません。

20.4.10.1 CSDO モード制御

無効な送信タイムスロットの間、CSDOM ビット (DCICON1<6>) の状態に応じて CSDO ピンは 0 に駆動するか、3 ステートにできます。TSCON レジスタで TSEx ビットをクリアすると、対応する送信タイムスロットは無効になります。

- CSDOM ビット (DCICON1<6>) がクリアされている場合 (既定値)、無効なタイムスロット周期の間、CSDO ピンは 0 に駆動します。シリアルバスに接続されているデバイスが 2 つ (マスタとスレーブ) しかない場合、このモードが使用されます。
- CSDOM ビット (DCICON1<6>) がセットされている場合、未使用のタイムスロット周期の間、CSDO ピンは 3 ステートです。このモードを使うと、多重化されたアプリケーションで複数の dsPIC33F デバイスが同じ CSDO ラインを共有することができます。CSDO ライン上の各デバイスは、特定のタイムスロットにのみデータを送信するよう設定されます。同じタイムスロットに 2 つのデバイスがデータを送信する事はできません。

20.4.11 受信スロット イネーブルビット

受信スロット イネーブル (RSCON) レジスタには、最大で 16 の受信用タイムスロットを有効化できる受信スロット イネーブル (RSE) 制御ビット (RSCON<15:0>) があります。各受信タイムスロットのサイズは、WS 制御ビット (DCICON2<3:0>) によって決まります (4 ～ 16 ビット)。

RSE ビットのいずれかによって受信タイムスロットを有効にする場合 (RSEx = 1)、シフトレジスタの内容が現在の DCI 受信バッファ位置に書き込まれ、バッファ制御ロジックは次に利用可能なバッファ位置にポインタを進めます。データを受信するには、少なくとも 1 つの受信タイムスロットが有効である必要があります。無効なタイムスロットが出現した場合、対応する RXBUFx レジスタの内容は受信されずにバッファポインタがインクリメントされます。

選択されたワードサイズが 16 ビット未満の場合、受信メモリバッファ位置の全てがデータで埋まりません。各受信スロットのデータワードは、それぞれ別の 16 ビットバッファ位置に格納されます。データは受信メモリバッファ内で常に左寄せで格納されます。従って、ワードサイズが 8 ビットの場合、受信データは RXBUFx レジスタのビット 15 からビット 8 に格納されます。

20.4.12 DCI バッファ制御ユニットの動作

DCI モジュールの働きにより、データの送信または受信処理中でも読み書き動作が可能です。データは TXBUFx レジスタに書き込まれ、RXBUFx レジスタから読み出されます。下図で、BLEN = 01 (バッファ長 = 2) の場合の内部 DCI 読み書き動作の例を説明します。

図 20-7 に、DCI モジュールが無効でデータが送受信されていない状態を示します。

図 20-7: DCI モジュール無効時

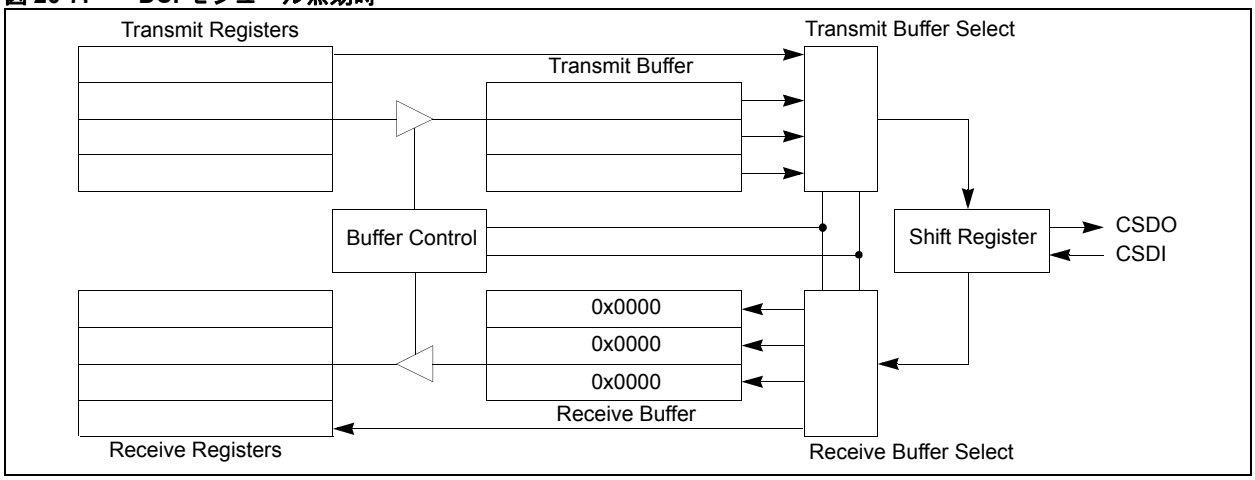
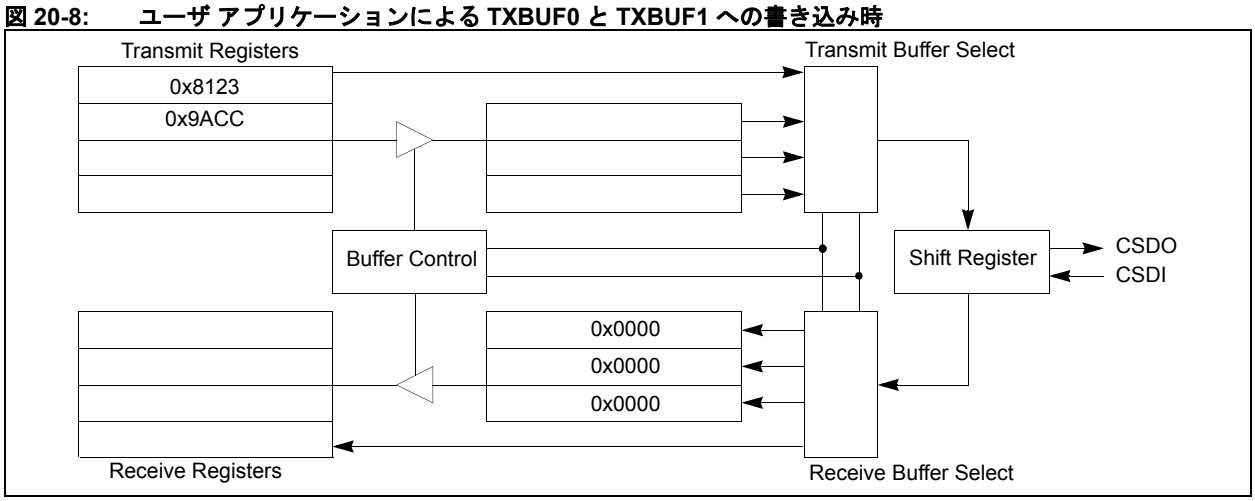


図 20-8 に、アプリケーションが TXBUF0 レジスタと TXBUF1 レジスタにデータを書き込んだ後の送信レジスタの状態を示します。

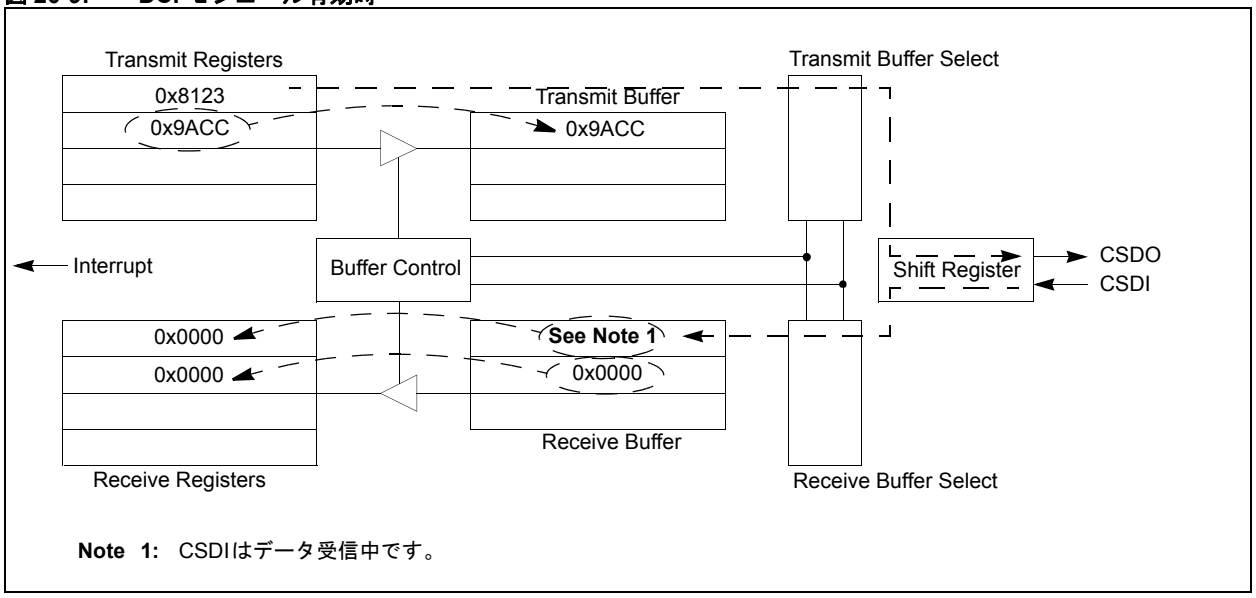
図 20-8: ユーザ アプリケーションによる TXBUF0 と TXBUF1 への書き込み時



DCI モジュールが有効になると、CPU は 3 クロックサイクル後に DCI 割り込みを受信します。この状況で、TXBUF0 内のデータはシフトレジスタに移動され、TXBUF1 内のデータは送信バッファに移動されます (図 20-9 参照)。DCI モジュールは CSDO ピンにデータ出力を開始します。

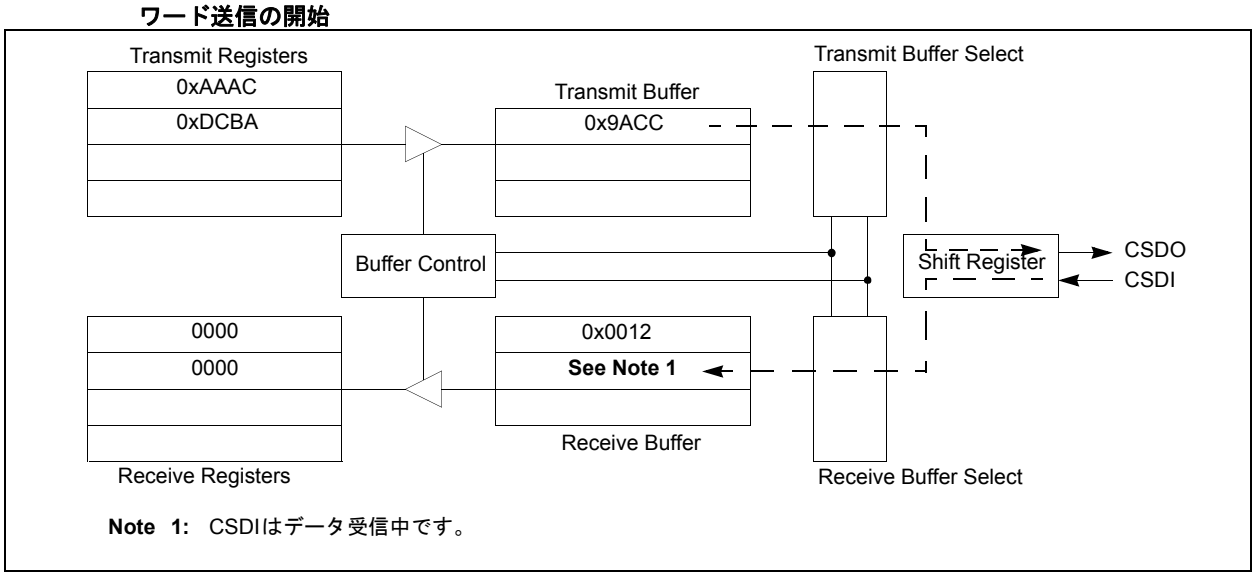
受信バッファの内容は RXBUF 受信レジスタに移動されます。モジュールが無効の時にはデータを受信しないため、これらの値は「0」として読み出されます。RXBUFx レジスタは次の割り込みが発生するまで「0」として読み出されます。次の割り込みの時点で受信バッファからのデータがこれらのレジスタに転送されます。モジュールは、CSDI ピンで受信したデータで受信バッファのデータの上書きを開始します。

図 20-9: DCI モジュール有効時



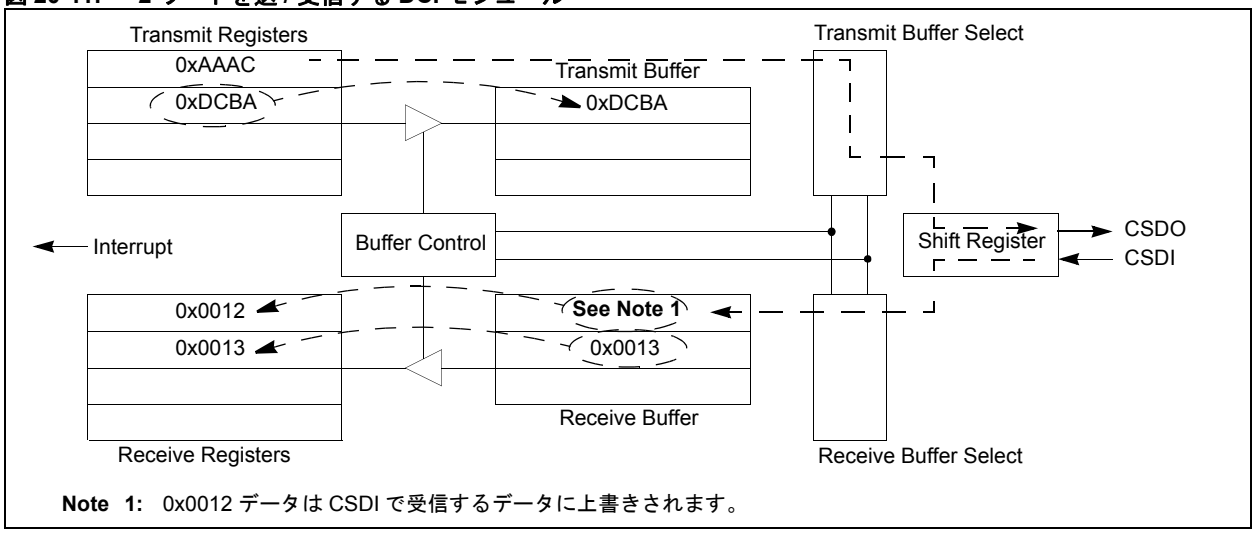
ユーザ アプリケーションは TXBUF0 と TXBUF1 に新しいデータを書き込みます。TXBUFx レジスタにデータを書き込んでも現在の送信動作に影響はありません。2 番目のデータワードはシフトレジスタに移動されます。図 20-10 に、新しいデータワードが CSDI ライン上で受信され受信バッファに移動される動きを示します。

図 20-10: ユーザ アプリケーションによる TXBUF0 と TXBUF1 への書き込み、DCI モジュールによる 2 番目のワード送信の開始



モジュールが BLEN + 1 ワードの送 / 受信動作を完了すると、割り込みが発生します。受信バッファのデータは RXBUFx レジスタにコピーされます。図 20-11 に、TXBUF0 内のデータがシフトレジスタに移動され、TXBUF1 レジスタの内容が送信バッファにコピーされる動きを示します。このサイクルは DCI 割り込みのたびに繰り返されます。

図 20-11: 2 ワードを送 / 受信する DCI モジュール



Note 1: 0x0012 データは CSDI で受信するデータに上書きされます。

20.4.13 バッファ制御ユニットを使用する TSCON と RSCON の動作

TSCON レジスタと RSCON レジスタのスロット イネーブルビットは、それぞれ独立して機能します (バッファ制御ロジックを除く)。データフレームの各タイムスロットでは、TSEx ビットと RSEx ビットのいずれか一方が現在のタイムスロットに設定されている場合、そのバッファ位置のポインタを先に進めます。つまり、バッファ制御ユニットは送受信バッファの同期をとるため、送受信のバッファ位置はデータフレーム内のタイムスロットで常に同じです。

データフレーム内で使用する全てのタイムスロットに TSEx ビットと RSEx ビットの両方を設定した場合、DCI が送信するデータ量と受信するデータ量は同じです。

アプリケーションによっては、1 フレームで送信するデータワード数と受信するワード数が一致しない場合があります。例えば、送信スロット 0 を有効化し (TSCON = 0x0001)、受信スロット 0 と 1 を有効化して (RSCON = 0x0003)、DCI を 2 ワードデータ フレームに設定したとします。DCI モジュールを 4 ワードで割り込みが発生するよう設定します (BLEN = 11)。フレームサイズは 2 データワードに設定 (COFSG = 0001)、データワード サイズは 8 ビットに設定します (WS = 0111)。図 20-12 にこの例におけるタイミング図、図 20-13 に対応する DCI バッファ動作を示します。この設定では、DCI は 1 フレームで 1 データワードを送信し、1 フレームで 2 データワードを受信します。1 データワードを送信するたびに 2 データワードを受信するため、ユーザ ソフトウェアは 1 つおきの送信バッファ位置に書き込みを行います。具体的には、データ送信に使用するのは TXBUF0 と TXBUF2 のみです。

図 20-12: TSCON = 0x0001 と RSCON = 0x0003 に設定した場合の DCI タイミング

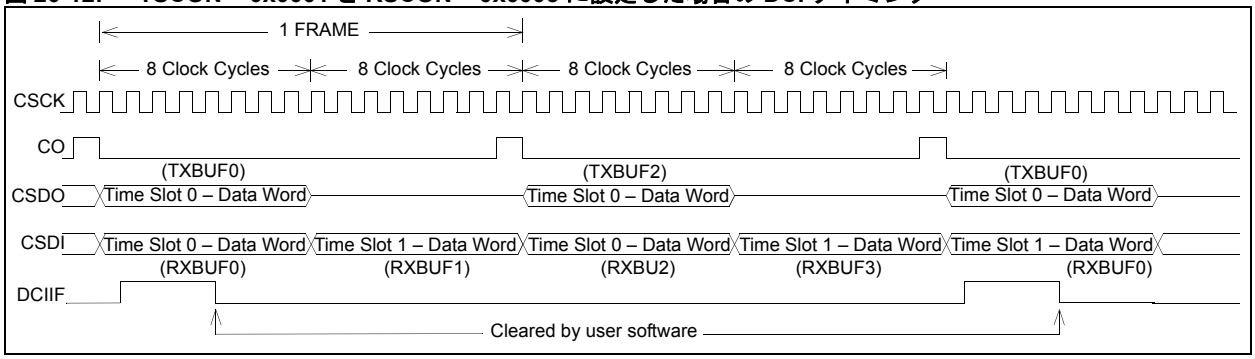
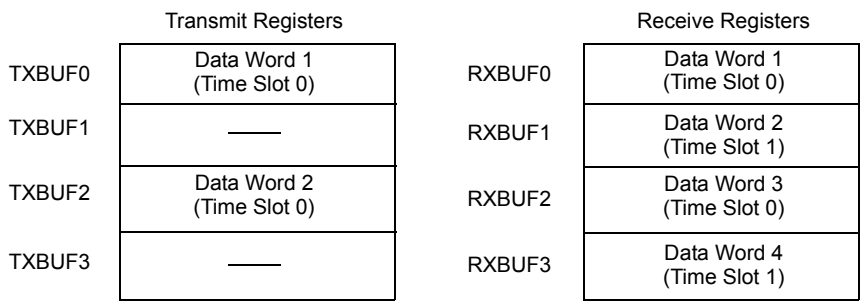


図 20-13: DCI バッファの動作 : TSCON = 0x0001, RSCON = 0x0003, BLEN<1:0> = 11



Note: ユーザ ソフトウェアは TXBUF0 と TXBUF2 に書き込みます。TXBUF1 と TXBUF3 は送信ロジックで使用されません。

BLEN + 1 バッファに書き込みが実行されて CPU 割り込みが発生した時、または TXBUF3 と RXBUF3 が処理されバッファポインタを最初のバッファ位置に戻す必要がある場合、バッファ制御は最初のバッファ位置を指すようリセットされます。

有効なタイムスロットの全ビットの処理が終了するかタイムスロットのビットが 16 ビットを超えた時点で、バッファ制御はバッファポインタをインクリメントします。TXBUF へのポインタは RXBUF ポインタと同期してインクリメントされます。

20.4.14 DMA を使用する TSCON と RSCON の動作

dsPIC33F の DMA モジュールを使うと、CPU に負荷をかけずに、デュアルポート SRAM と DCI TXBUF0/RXBUF0 レジスタ間で直接データを転送する事ができます。正常に動作させるには、BLEN ビット (DCICON2<11:10>) を「0」に設定します。DCI モジュールはこのモードでの動作に TXBUF0 と RXBUF0 のみを使用しますが、マルチワード フレームと複数タイムスロットを利用する事も可能です。ユーザ アプリケーションは、メモリに格納されるデータを確実に有効なタイムスロットに対応させる必要があります。

図 20-12 は DCI コーデック通信の例です。ここでは DCI モジュールに、有効な送信タイムスロットが 1 つ (TS0)、有効な受信タイムスロットが 2 つ (RS0 と RS1) あります。ワード長は 8 ビット (WS = 0111) でフレームサイズは 2 ワード (COFSG = 0001) です。BLEN = 0 で、DCI モジュールは各ワードで DMA 転送を要求します。DCI モジュールは 8 ビット ワードサイズに設定されており、データの MSb を最初に送信します。従って、送信される 8 ビットデータが 16 ビットワードの最上位バイト (MSB) に格納されるよう DPSRAM データを構成する必要があります。図 20-12 に示すタイミング基準を満たすには、1 つおきのワードが送信対象のデータとなるように、DPSRAM の送信データメモリに追加設定が必要です。

- **転送 1**
  DMA モジュールは、DPSRAM の内容を TXBUF0 に、RXBUF0 の内容を DPSRAM に格納します。TXBUF0 と RXBUF0 のデータはタイムスロット 0 に対応します。DMA ポインタがインクリメントされます。データワード サイズが 8 ビットである事から、受信データは DPSRAM ワードの上位 8 ビットに格納されます (図 20-14 参照)。
- **転送 2**
  DMA モジュールは、DPSRAM の内容を TXBUF0 に、RXBUF0 の内容を DPSRAM に格納します。このデータはタイムスロット 1 に対応します。送信タイムスロット 1 は無効である事から、DCI モジュールはデータを送信しません。しかし受信タイムスロット 1 は有効であるため、RXBUF0 レジスタには CSDI ピンで受信したデータを格納します (図 20-15 参照)。データは DPSRAM に格納され、DMA ポインタがインクリメントされます。
- **転送 3**
  フレーム長が 2 ワードであるため、DCI モジュールは COFS 信号を生成します。DMA モジュールは DPSRAM の内容を TXBUF0 に、RXBUF0 の内容を DPSRAM に格納します。TXBUF0 と RXBUF0 のデータはタイムスロット 0 に対応します。DMA ポインタがインクリメントされます。データワード サイズが 8 ビットであるため、受信データは DPSRAM ワードの上位 8 ビットに格納されます (図 20-16 参照)。
- **転送 4**
  DMA モジュールは DPSRAM の内容を TXBUF0 に、RXBUF0 の内容を DPSRAM に格納します。このデータはタイムスロット 1 に対応します。送信タイムスロット 1 は無効であるため、DCI モジュールはデータを送信しません。しかし受信タイムスロット 1 は有効であるため、RXBUF0 レジスタには CSDI ピンで受信したデータを格納します (図 20-17 参照)。データは DPSRAM に格納され、DMA ポインタがインクリメントされます。

図 20-14:　転送 1: タイムスロット 0

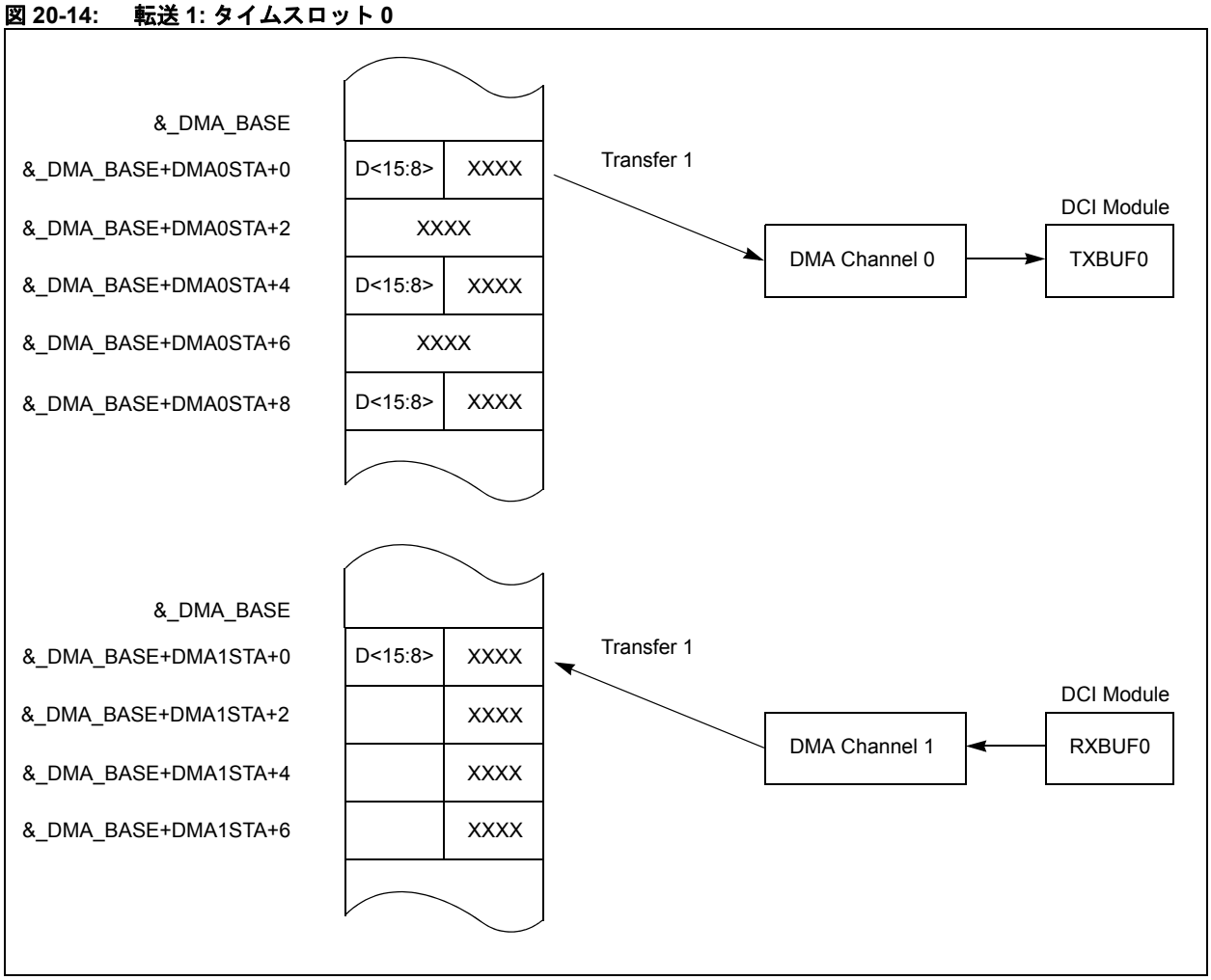


図 20-15:　転送 2: タイムスロット 1

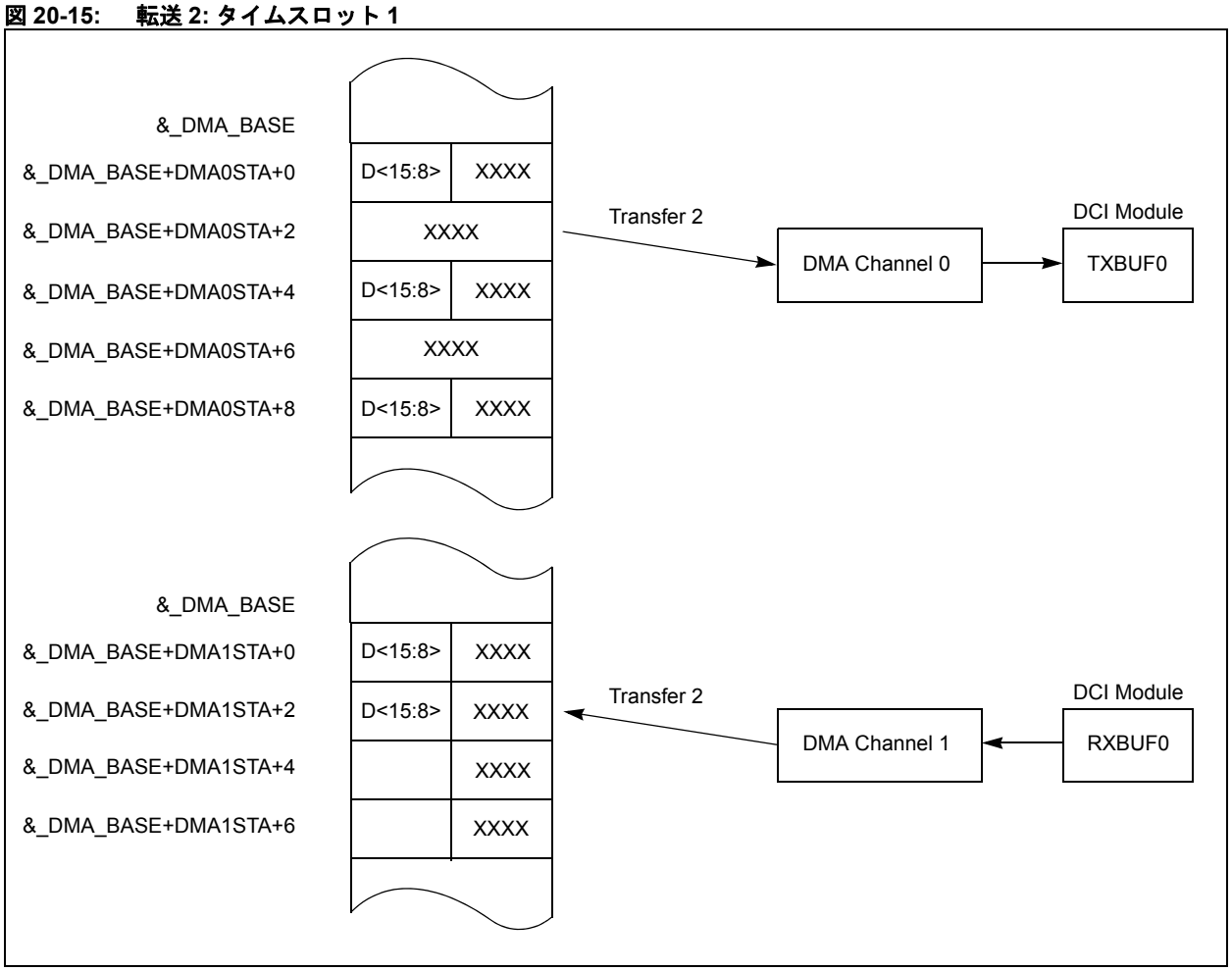


図 20-16: 転送 3: タイムスロット 0

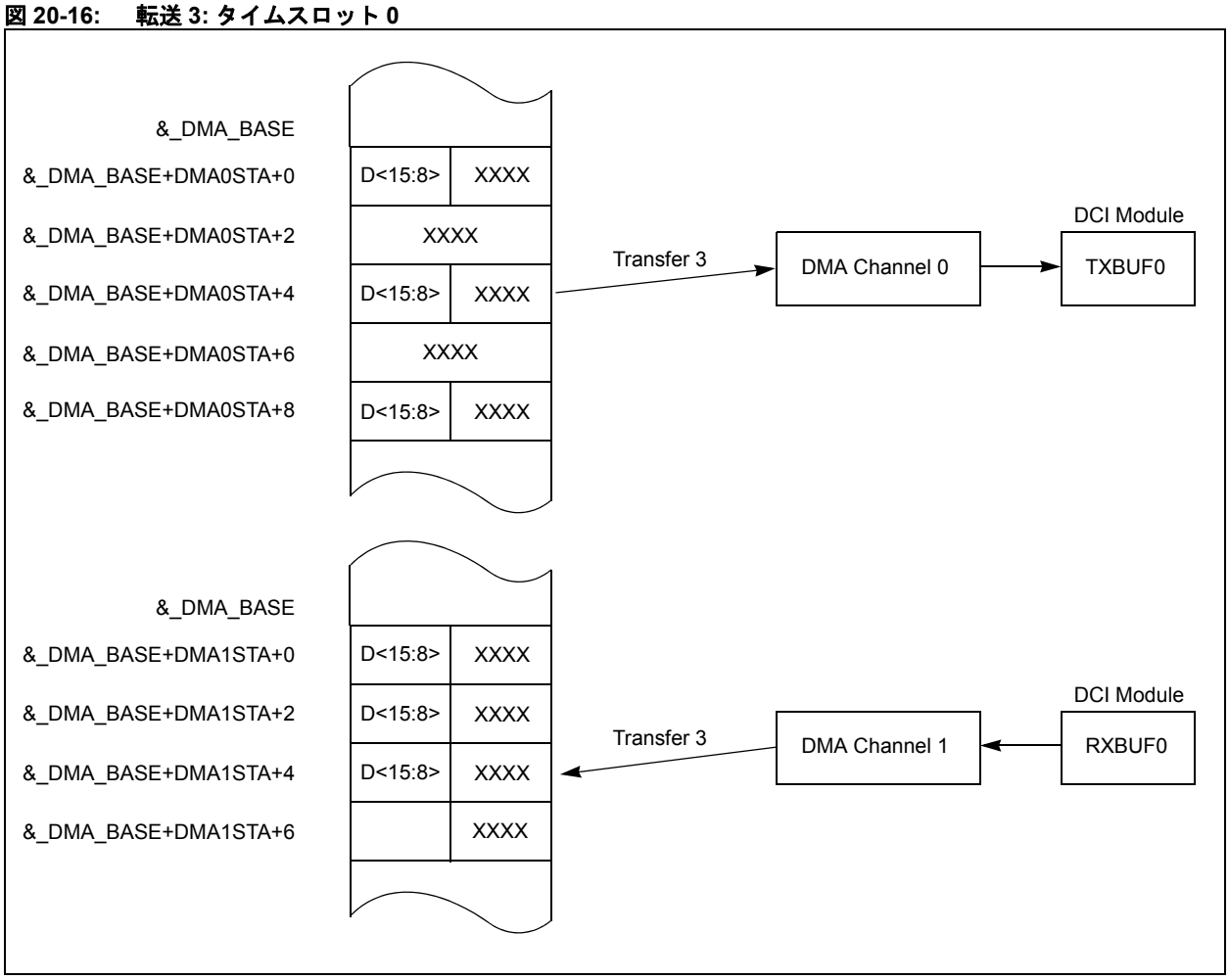
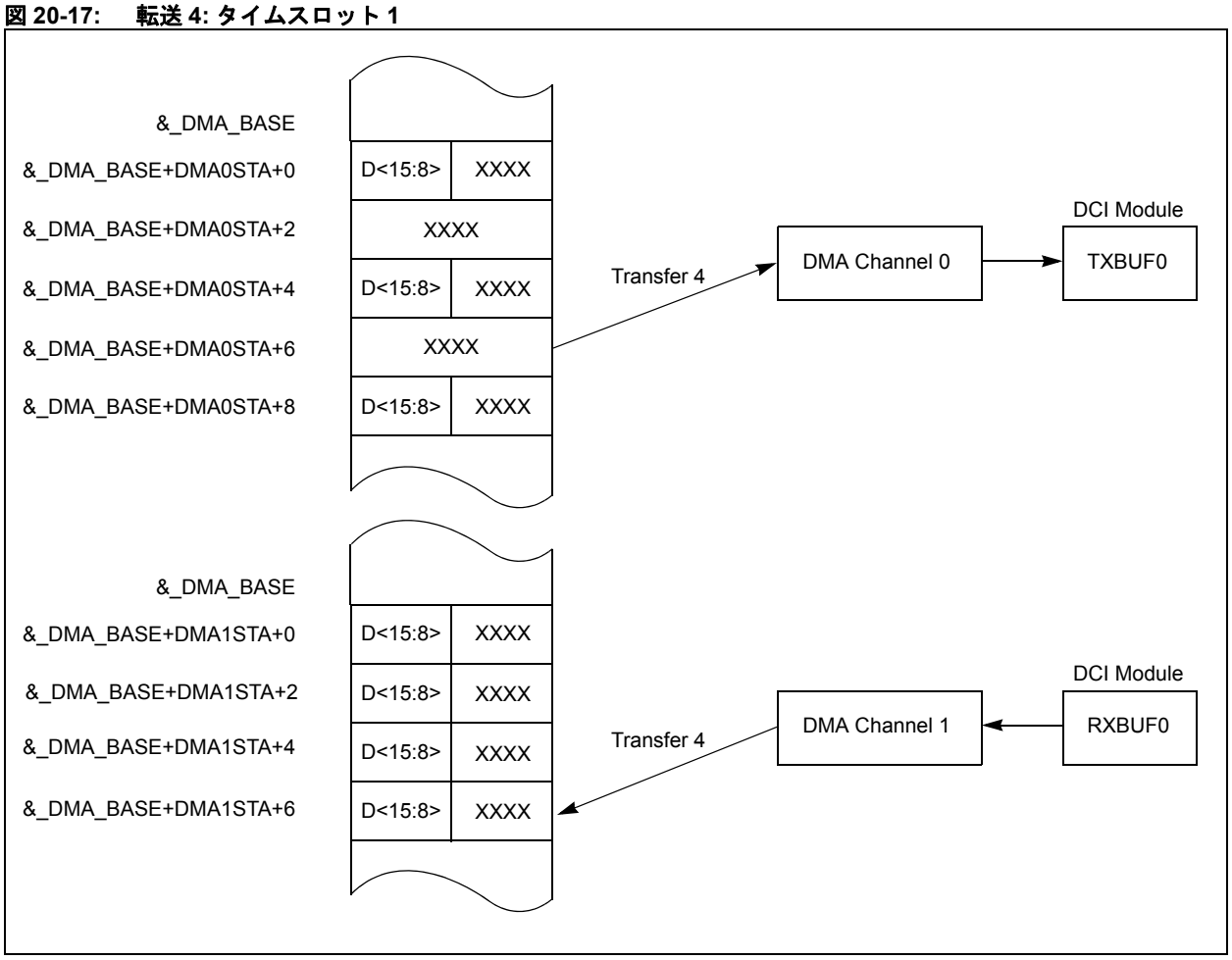


図 20-17:　転送 4: タイムスロット 1



20.4.15　受信ステータスビット

受信バッファフル (RFUL) と受信オーバーフロー(ROV) という 2 つの受信ステータスビットは、モジュールが使用するために有効になっているレジスタ位置のステータスのみを示します。これは BLEN 制御ビット (DCICON2<11:10>) の機能です。バッファ長が 4 ワード未満に設定されている場合、未使用のバッファ位置は受信ステータスビットに影響を与えません。

RFUL ステータスビット (DCISTAT<2>) は読み出し専用で、受信レジスタに新しいデータがある事を示します。RFUL ビットは、使用中の全ての RXBUF レジスタがユーザ ソフトウェアによって読み出された時点で自動的にクリアされます。

ROV ステータスビット (DCISTAT<3>) は読み出し専用で、受信レジスタ位置のどれかで受信オーバーフローが発生した事を示します。受信オーバーフローは、新しいデータがバッファメモリから転送される前に、ユーザ ソフトウェアによって RXBUF レジスタ位置が読み出されていない場合に発生します。受信オーバーフローが発生すると、レジスタの既存の内容は上書きされます。ROV ステータスビットは、オーバーフローを引き起こしたレジスタが読み出された時点で自動的にクリアされます。

20.4.16 送信ステータスビット

送信バッファ エンプティ (TMPTY) と送信バッファ アンダーフロー (TUNF) という 2 つの送信ステータスビットは、モジュールが使用するレジスタ位置のステータスのみを示します。バッファ長が 4 ワード未満に設定されている場合、未使用のレジスタ位置は送信ステータスビットに影響を与えません。

TMPTY ビット (DCISTAT<0>) は読み出し専用で、アクティブな TXBUF レジスタの内容が送信バッファ レジスタに転送された時点でセットされます。送信レジスタが書き込み可能かどうか判断するため、TMPTY ビットをソフトウェアでポーリングできます。TMPTY ビットは、使用中の TXBUF レジスタのいずれかへの書き込みが発生した時点でハードウェアによって自動的にクリアされます。

TUNF ビット (DCISTAT<1>) は読み出し専用で、使用中の送信レジスタのいずれかで送信アンダーフローが発生した事を示します。TUNF ビットは、TXBUF レジスタの内容が送信バッファ メモリに転送され、ユーザ ソフトウェアが直近のバッファ転送以降に使用中の全 TXBUF レジスタを書き込んでいない場合にセットされます。TUNF ステータスビットは、アンダーフローが発生した TXBUF レジスタがユーザ ソフトウェアによって書き込まれた時点で自動的にクリアされます。

20.4.17 SLOT ステータスビット

SLOT ステータスビット (DCISTAT<11:7>) は、データフレーム内の現在アクティブなタイムスロットを示します。これらのビットは、データフレームごとに 4 ワードを超えるデータを転送する必要がある場合に便利です。ユーザ ソフトウェアは、DCI 割り込み発生時にどのタイムスロットのデータを最後に受信したか、どのタイムスロットのデータを TXBUF レジスタに読み込むべきかを判断するため、このステータスビットをポーリングできます。

20.4.18 デジタル ループバック モード

デジタル ループバック モード (DLOOP) 制御ビット (DCICON1<11>) をセットすると、デジタル ループバック モードを有効にできます。DLOOP ビットがセットされると、モジュールは内部的に CSDO 信号を CSDI に接続します。デジタル ループバック モードでは、CSDI ピンの実際のデータ入力は無視されます。

20.4.19 アンダーフロー モード制御ビット

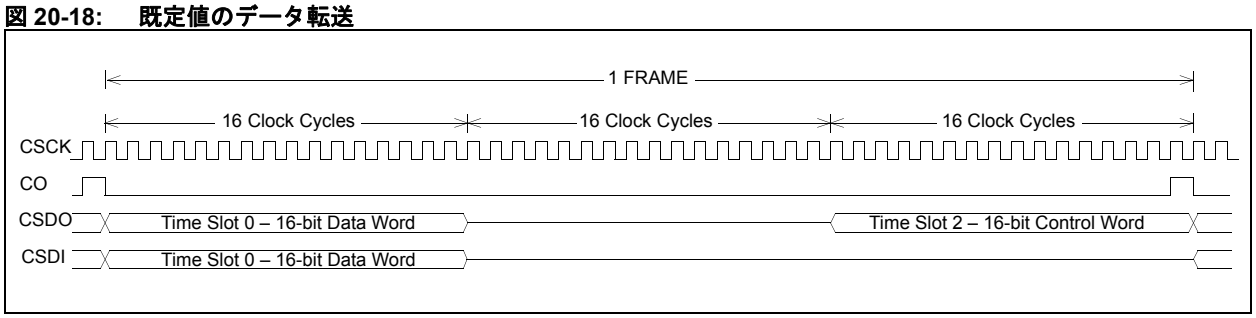
送信アンダーフローが発生すると、アンダーフロー モード (UNFM) 制御ビット (DCICON1<7>) の状態によって、以下の 2 つの動作のいずれかが発生します。

- UNFM ビットがクリアされている場合 (既定値) 、モジュールは、バッファ位置用の動作タイムスロット中に CSDO ピンに 0 を送信します。この動作モードでは、DCI モジュールに接続されたコーデックデバイスにデジタル的な「静寂」が与えられます。
- UNFM 制御ビットがセットされている場合、モジュールはバッファ位置に書き込まれた最新データを送信します。この動作モードでは、ユーザ ソフトウェアはソフトウェアにオーバーヘッドをかけずに連続したデータ値をコーデック デバイスに送信できます。

20.4.20 データ位置調整制御

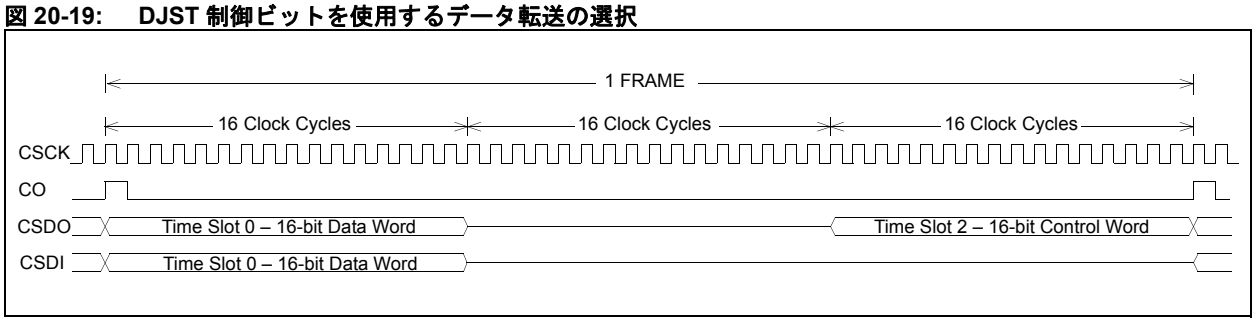
ほとんどのアプリケーションでは、アクティブな FS 信号がサンプリングされてから 1 シリアルクロック サイクル後にデータ転送を開始します (図 20-18 参照)。これは DCI モジュールの既定値です。

図 20-18: 既定値のデータ転送



別のデータ位置調整を選択するには、DJST 制御ビット (DCICON1<5>) を設定します。DJST = 1の場合､FS信号と同じシリアルクロック サイクル中にデータ転送が始まります(図20-19参照)。

図 20-19: DJST 制御ビットを使用するデータ転送の選択



20.4.21 DCI モジュールの割り込み

DCI モジュール割り込みの頻度は、BLEN 制御ビットによって決まります。バッファ長の値に達すると割り込みが発生します。DCI モジュールが有効になる前に割り込みが有効になると、モジュールが有効になってから 3 CSCK サイクル後に割り込みが発生します。

DCI モジュールにはエラー割り込み機能もあります。エラー割り込みを有効にすると、送信アンダーフロー イベントまたは受信オーバーフロー イベントが発生した時点で CPU 割り込みが発生します。

20.5 DCI モジュールの使用

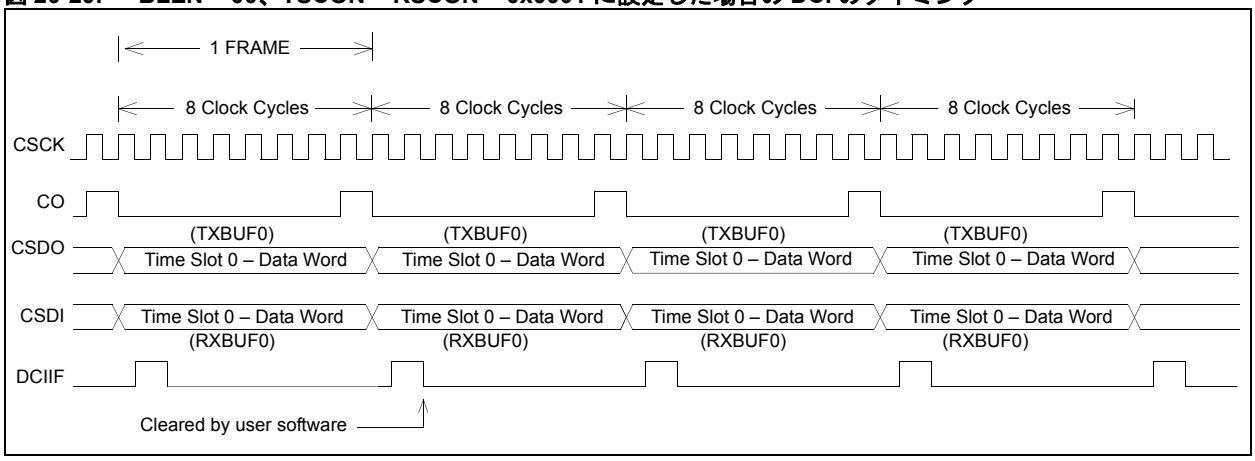
本セクションでは、特定の種類のデータコンバータを使用する DCI の設定方法と使用方法を説明します。

20.5.1 DCI バッファ、ステータスビット、割り込みを使用したデータ送受信方法

DCI は、BLEN 制御ビットの設定に応じて、CPU 割り込み間にデータワードを最大 4 つまでバッファできます。バッファされたデータは、TSCON レジスタと RSCON レジスタの設定に応じて、1 つまたは複数のデータフレームで送受信が可能です。以下に 4 種類の設定例を挙げます。

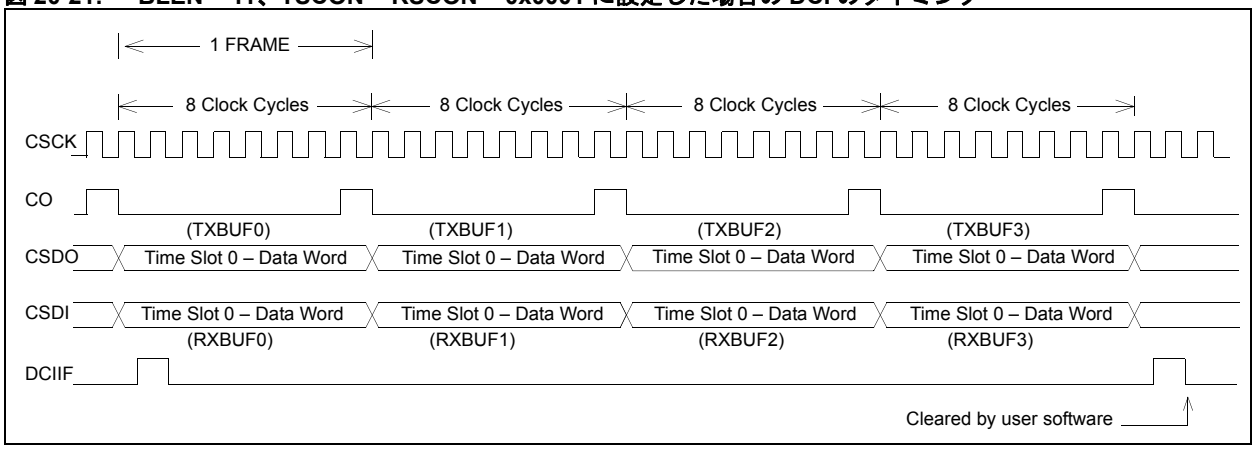
1. BLEN<1:0> = 00 (割り込みごとに 1 データワードをバッファ)、TSCON = RSCON = 0x0001 と想定します。これは最も基本的な設定であり、各データフレームの開始時点で DCI が 1 データワードを送受信します。BLEN<1:0> = 00 という設定から、1 データワードを送受信するたびに CPU 割り込みが発生します。詳細は、図 20-20 を参照してください。

図 20-20: BLEN = 00、TSCON = RSCON = 0x0001 に設定した場合の DCI のタイミング



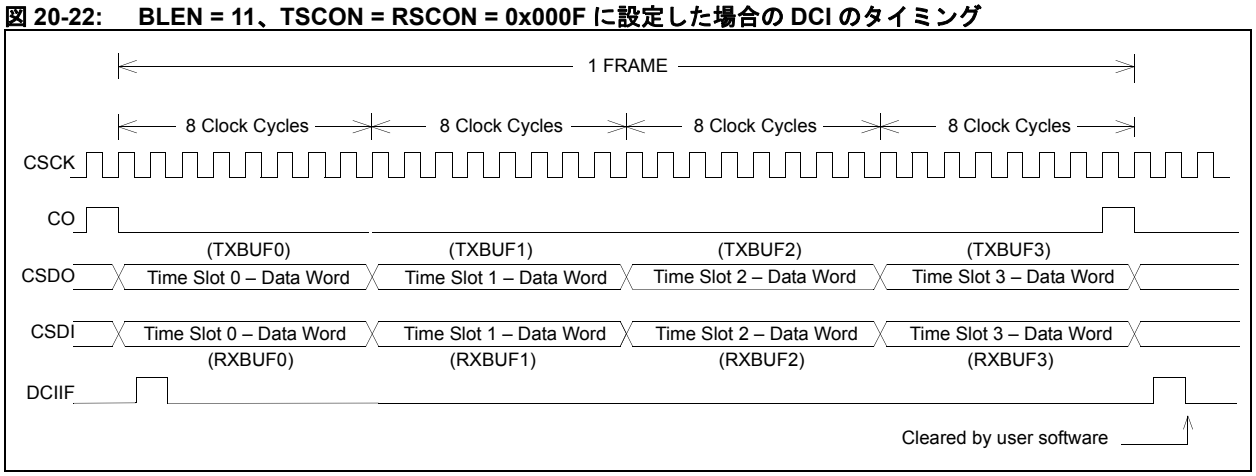
2. BLEN<1:0> = 11 (割り込みごとに 4 データワードをバッファ)、TSCON = RSCON = 0x0001 と想定します。この設定では、各データフレームの開始時点で DCI が 1 データワードを送受信します。4 データワードを送受信した後に CPU 割り込みが発生します。この設定はブロック処理に便利です。複数のデータサンプルをまとめて処理します。詳細は、図 20-21 を参照してください。

図 20-21: BLEN = 11、TSCON = RSCON = 0x0001 に設定した場合の DCI のタイミング



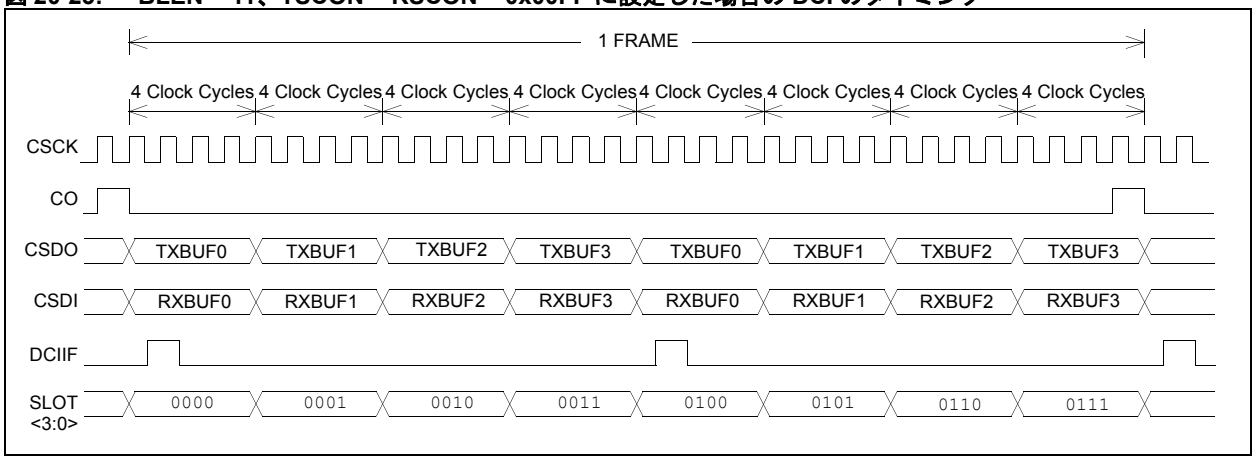
3. BLEN<1:0> = 11 (割り込みごとに 4 データワードをバッファ)、TSCON = RSCON = 0x000F と想定します。この設定では、各データフレームの開始時点で DCI が 4 データワードを送受信します。この場合、1 つのデータフレーム内に 4 データワードをバッファするよう DCI を設定しているため、各データフレームで CPU 割り込みが発生します。これはマルチチャンネル バッファリングの代表的な設定です。詳細は、図 20-22 を参照してください。

図 20-22: BLEN = 11、TSCON = RSCON = 0x000F に設定した場合の DCI のタイミング



4. DCI がフレームごとに 5 つ以上のデータワードをバッファするように設定する事もできます。例えば、BLEN<1:0> = 11 (割り込みごとに 4 データワードをバッファ)、TSCON = RSCON = 0x00FF と想定します。この設定では、DCI はデータフレームごとに 8 データワードを送受信します。データフレームごとに割り込みが 2 回発生します。割り込みのたびにデータのどの部分が送信レジスタまたは受信レジスタにあるかを判断するために、ユーザ ソフトウェアは割り込みサービスルーチン (ISR) の SLOT ステータスビット (DCISTAT <11:7>) をチェックして現在のデータフレーム位置を特定する必要があります。図 20-23 に、この場合の 4 ビットの例を示します。

図 20-23: BLEN = 11、TSCON = RSCON = 0x00FF に設定した場合の DCI のタイミング



送信レジスタと受信レジスタはダブルバッファ構造であるため、ユーザ ソフトウェアが一方のセットの送受信データを操作している間に、DCI モジュールが他方のセットの送受信データを操作できます。ダブルバッファ構造ではデータの受信、データの処理、処理済みデータの送信に 3 つの割り込み周期を必要とします。CPU は DCI 割り込みごとに前回の割り込み周期で受信したデータワードを処理し、次回の割り込み周期で送信するデータワードを生成します。dsPIC DSC デバイスのバッファリングとデータ処理の時間のため、処理されるデータに割り込み周期 2 回分の遅延が加えられます。ほとんどの場合、このデータの遅延は無視できる程度です。

DCI ステータスフラグと CPU 割り込みは、バッファ転送が実行され CPU が次のデータを処理するタイミングである事を示します。代表的なアプリケーションでは、DCI データが処理されるたびに以下のように動作します。

1. ユーザ ソフトウェアが RXBUF レジスタを読み出します。
2. モジュールが RFUL ステータスビット (DCISTAT<2>) をセットし、受信レジスタに新しいデータがある事を示します。
3. RFUL ビットはアクティブな受信レジスタが全て読み出された後、自動的にクリアされます。
4. ユーザ ソフトウェアが受信データを処理します。
5. TMPTY ステータスビット (DCISTAT<0>) がセットされ、送信レジスタに次のデータを書き込める状態である事を示します。
6. 処理済みのデータが TXBUF レジスタに書き込まれます。

データを送受信するように設定されているアプリケーション (TSCON と RSCON が 0 以外) では、ユーザ ソフトウェアで RFUL (DCISTAT<2>) ステータスビットと TMPTY (DCISTAT<0>) ステータスビットをポーリングして、DCI バッファ転送を実行するタイミングを判断できます。

- DCI がデータ送信専用の場合 (RSCON = 0)、バッファ転送を示すために TMPTY ビットをポーリングします。
- DCI がデータ受信専用の場合 (TSCON = 0)、バッファ転送を示すために RFUL ビットをポーリングします。

DCIIF ステータスビット (IFS2<9>) は DCI バッファ転送が実行されるたびにセットされ、有効な場合には CPU 割り込みが発生します。DCIIF ステータスビットは、RFUL (DCISTAT<2>) ステータスビットと TMPTY (DCISTAT<0>) ステータスビットを論理 OR 演算する事によって生成されます。

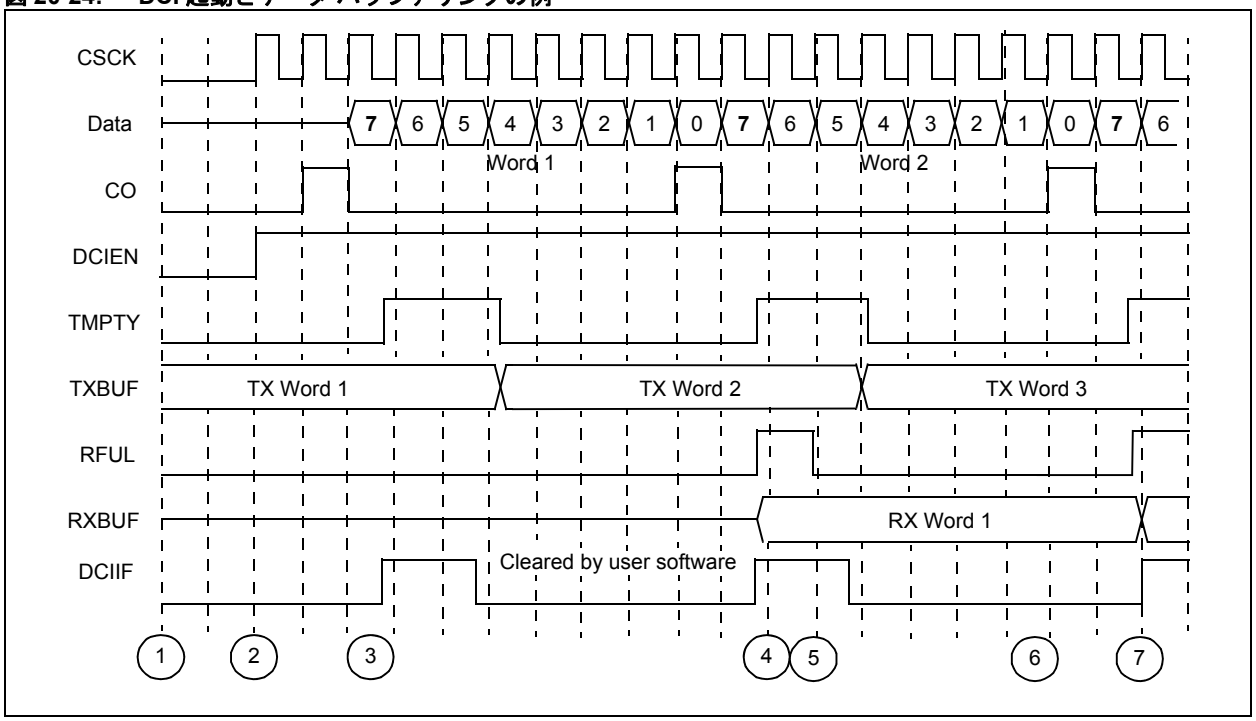
20.5.1.1 DCI 起動とデータ バッファリング

DCI 起動では、最初に所定の動作モードに対応して DCI 制御レジスタを初期化します。データ転送を開始するには、DCIEN 制御ビット (DCICON1<15>) をセットします。**20.5.4「マルチチャンネルの動作」**、**20.5.5「I2S の動作」**、**20.5.6「AC-LINK の動作」**を参照してください。

図 20-24 に、DCI 起動のタイミング図を示します。この例では、DCI は 8 ビット データワード (WS<3:0> = 0111) と 8 ビットデータフレーム (COFSG<3:0> = 0000) 用に設定されています。バッファ長は 1 (BLEN = 00) に設定されており、送信タイムスロット 0 (TSCON = 0x1) と、受信タイムスロット 0 (RSCON = 0x1) が有効です。さらに、マルチチャンネル モード (COFSM<1:0> = 00) を使用します。データ送受信の手順は以下の通りです。

1. モジュールを有効にする前に、最初に送信するデータを格納した TXBUF レジスタをプリロードします。送信データがコーデックから受信したデータに基づく場合、ユーザ ソフトウェアは単に TXBUF レジスタをクリアするだけです。コーデックから RXBUF レジスタに最初のデータを受信するまでは、デジタルな「無信号」が送信されます。
2. DCI モジュールを有効にするには、DCIEN ビット (DCICON1<15>) を設定します。DCI がマスタデバイスである場合、TXBUF レジスタ内のデータが送信バッファに転送され、最初のデータフレーム伝送が開始します。DCI がマスタデバイスでない場合、マスタデバイスから FS 信号を受信するまで TXBUF データは送信バッファに保持されます。
3. TMPTY ビット (DCISTAT<0>) はモジュールが有効になった 3 クロックサイクル後にセットされ、DCI 割り込みが発生します (有効な場合)。この時点でモジュールは、2 番目のデータフレームで転送するデータを TXBUF レジスタに再読み込みする準備が整っています。モジュールはデータを全く受信していないため、転送データを受信データから計算する場合、TXBUF レジスタを再度クリアします。割り込みが有効な場合、ユーザ ソフトウェアの DCIIF ステータスビットをクリアします。
4. 最初のデータフレームの転送後、TMPTY ビット (DCISTAT<0>) がセットされ、RFUL ステータスビットがセットされ、DCI 割り込みが発生します (有効な場合)。これは、DCI に接続されているデバイスから受信する最初のデータワードです。
5. ユーザ ソフトウェアが受信レジスタを読み出し、RFUL ステータスビットが自動的にクリアされます。ユーザ ソフトウェアは受信データの処理も行います。
6. 送信レジスタに、次のデータフレームで送信するデータが書き込まれます。TMPTY ステータスビット (DCISTAT<0>) は、書き込みが発生すると自動的にクリアされます。書き込みデータは、前回の割り込みで受信したデータから計算できます。
7. 次の DCI 割り込みが発生しサイクルが繰り返されます。

図 20-24:　DCI 起動とデータ バッファリングの例



20.5.1.2　DCI の無効化

DCI モジュールを無効にするには、DCIEN 制御ビット (DCICON1<15>) をクリアします。DCIEN ビットがクリアされると、モジュールは処理中のデータフレーム転送を終了します。フレームの終了前に送受信バッファを読み書きする必要がある場合、割り込みが発生します。

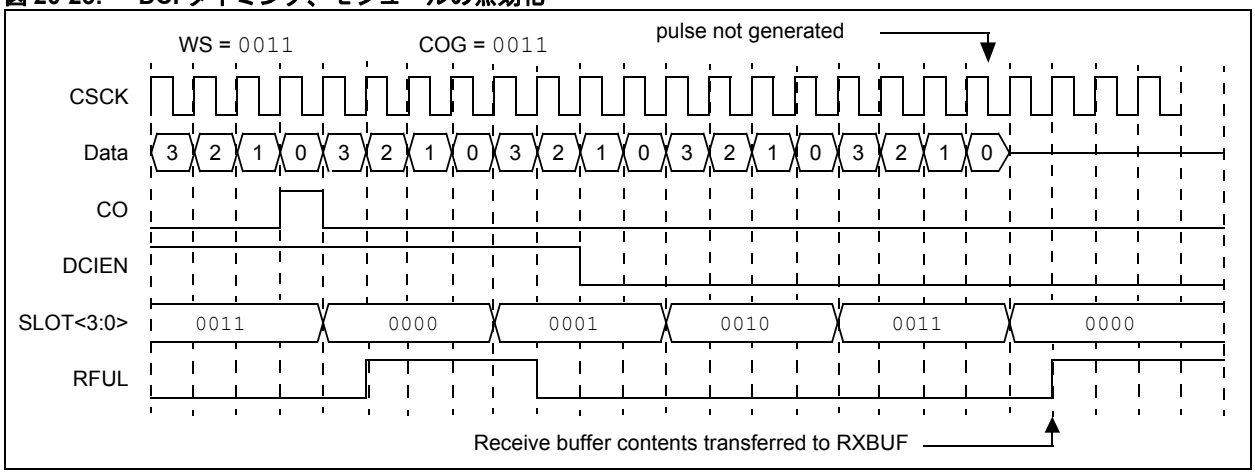
フレーム内でモジュールを無効にするには、フレームが終了する少なくとも 3 CSCK サイクル前に DCIEN ビットをクリアする必要があります。このタイミングに遅れてビットをクリアした場合、モジュールは次のフレームで無効になります。

DCIが無効化されると、それ以降FSパルスを生成せず、入力FSパルスにも応答しなくなります。

FSG がデータフレーム内の最後のタイムスロットに達すると、DCI に関連付けられている全てのステートマシンがアイドル ステートにリセットされ、モジュールに関連する I/O ピンの制御が解放されます。モジュールがアイドルになっているタイミングを判別するために、DCIEN ビット (DCICON1<15>) がクリアされた後、ユーザ ソフトウェアは SLOT ステータスビット (DCISTAT<11:8>) をポーリングできます。SLOT<3:0> = 0000、DCIEN = 0 の場合、DCI はアイドル ステートです。

モジュールがアイドルステートに入ると、受信シャドウレジスタ内のデータはすべて RXBUF レジスタに転送され、必要に応じて RFUL と ROV ステータスビットも変更されます。

図 20-25: DCI タイミング、モジュールの無効化



20.5.2 マスタ動作とスレーブ動作

DCI はマスタ動作またはスレーブ動作に設定できます。マスタデバイスは FS 信号を生成してデータ転送を開始します。動作モード (マスタまたはスレーブ) は COFSD 制御ビット (DCICON1<8>) により選択します。

DCI モジュールがマスタデバイスとして動作する場合 (COFSD = 0)、COFSM モードビット (DCICON1<1:0>) によって、FSG ロジックが生成する FS パルスのタイプが決まります。FSG がリセットされると新たな FS 信号が生成され、COFS ピンに出力されます。

DCI モジュールが FS スレーブとして動作する場合 (COFSD = 1)、DCI モジュールに接続されているデバイスがデータ転送を制御します。COFSM 制御ビット (DCICON1<1:0>) は、入力 FS 信号に対する DCI モジュールの応答方法を制御します。

マルチチャンネル モードでは、COFS ピンで High がサンプリングされた 1 シリアルクロック サイクル後に新しいデータフレーム転送が開始されます。FSG ロジックは COFS ピンのパルスによってリセットされます。

I^2S モードでは、COFS ピンで Low から High または High から Low への遷移がサンプリングされた 1 シリアルクロック サイクル後に新しいデータワードが転送されます。FSG ロジックは COFS ピンの立ち上がりエッジまたは立ち下がりエッジでリセットされます。

AC-LINK モードでは、COFS ピンで High がサンプリングされた 1 シリアルクロック サイクル後に次のフレームのタグスロットとこれに続く複数のデータスロットが転送されます。

COFSM (DCICON1<1:0>) ビットと WS (DCICON2<3:0>) ビットは、モジュールがスレーブモードで動作する場合、予測されるフレーム長を示すために設定する必要があります。モジュールが COFS ピンで有効な FS パルスをサンプリングすると、データフレーム全体の転送が実行されます。モジュールは現在のデータフレーム転送が全て完了するまで、後続の FS パルスに応答しません。

20.5.3 ロングワード データ サポートのデータ パッキング

多くのコーデックのデータワード長は 16 ビットを超えています。DCI は最大 16 ビットまでのワード長を標準でサポートしていますが、それ以上長いワード長をサポートするには複数の送受信スロットを有効にし、データを複数の送受信バッファ位置にパッキングします。

例えば、あるコーデックが 24 ビット データワードを送信または受信すると想定します。このデータを送受信するには、BLEN<1:0> = 01 (割り込みごとに 2 データワード)、TSCON = 0x0003、RSCON = 0x0003 に設定します。この設定により、データフレームの最初の 2 つのタイムスロットで送受信が可能になります。送信データの 16 MSb は TXBUF0 に書き込まれます。送信データの 8 LSb は、図 20-26 に示すように左寄せで TXBUF1 に書き込まれます。TXBUF1 の 8 LSb の値には「0」を書き込む事ができます。コーデックから受信する 24 ビットデータは、送信データと同じフォーマットで RXBUF0 と RXBUF1 に読み込まれます。この場合、FS 信号は 32 ビット間隔で生成されます。

複数の送受信レジスタにあるロングワード データを送受信するために、ワードサイズと有効なタイムスロットをどのように組み合わせる事もできます。例えば図 20-26 に示す 24 ビット データワードは、WS<3:0> = 0111 (ワードサイズ = 8 ビット)、BLEN<1:0> = 10 (割り込み間に 3 つのワードをバッファ)、TSCON = RSCON = 0x0007 (データフレームの最初の 3 つのタイムスロットで送受信) と設定する事によって連続する 3 つのレジスタで送受信できます。各送受信レジスタは 8 ビットのデータワードを格納します (図 20-27 参照)。COFSG = 0010 (フレームごとに 3 ワード) の場合、FS 信号は 24 ビット間隔で生成されます。

図 20-26: ロングワード データのデータ パッキングの例

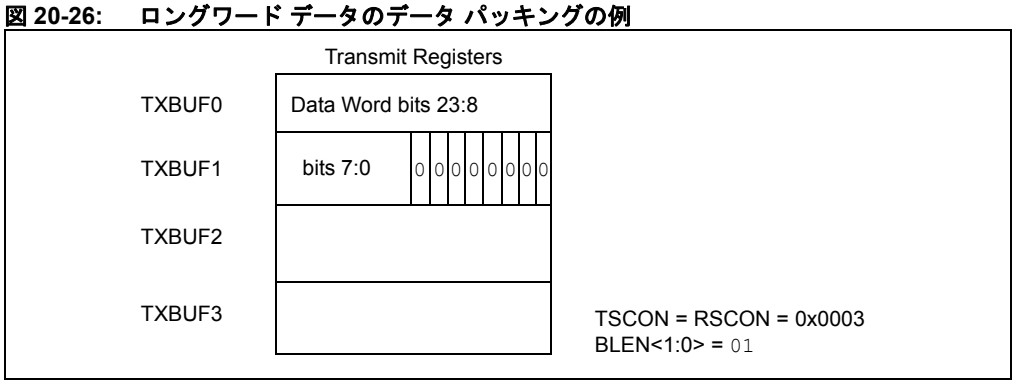
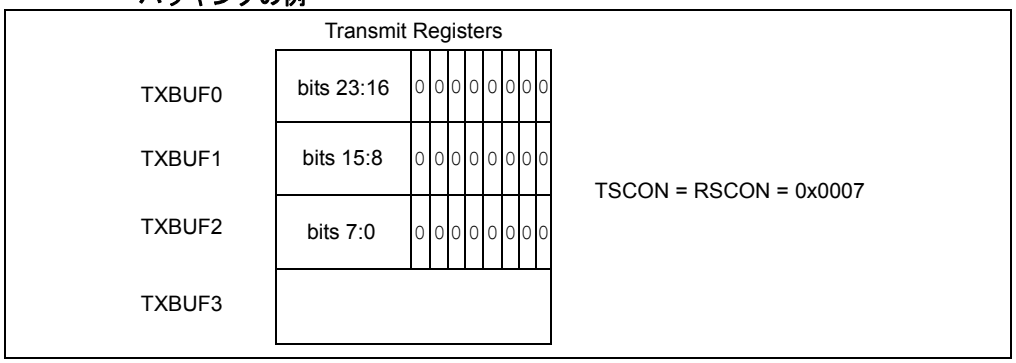


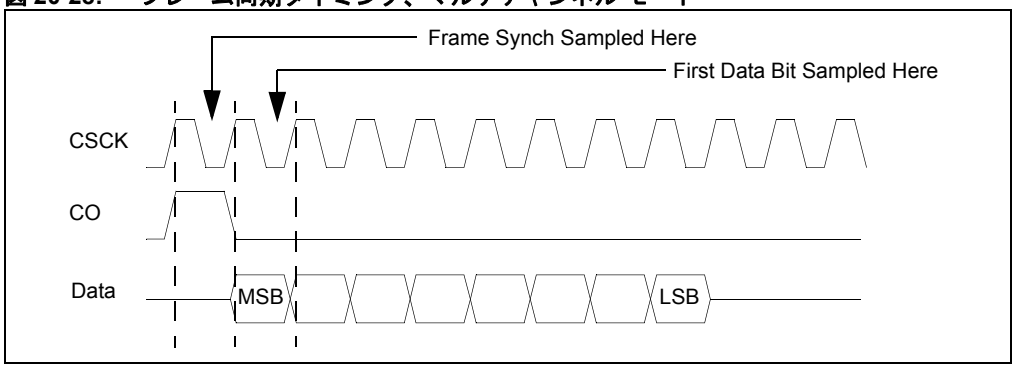
図 20-27: WS = 0111、TSCON = RSCON = 0x0007 の場合のロングワード データのデータ パッキングの例



20.5.4 マルチチャンネルの動作

マルチチャンネル モード (COFSM<1:0> = 00) は、データ転送を開始するために 1 シリアルクロック周期の間 FS パルスを High に駆動する必要があるコーデックで使用します。データフレームで 1 つまたは複数のデータワードを転送できます。連続する FS パルス間のクロックサイクル数は、DCI モジュールに接続されているデバイスによって異なります。図 20-28 はマルチチャンネル モードの FS 信号のタイミング図です。図 20-2 は 4 ビット ワードデータ転送を示すタイミングの例です。

図 20-28: フレーム同期タイミング、マルチチャンネル モード



20.5.4.1 マルチチャンネルのセットアップの詳細

本セクションでは、マルチチャンネル モードを使用してコーデック用に DCI を設定するために必要な手順を説明します。この動作モードは 1 つまたは複数のデータ チャンネルを持つコーデックに適用できます。チャンネル数に関わらず設定方法は同様です。

この例では、仮想のコーデックを使用します。ここで取り上げるシングル チャンネル コーデックは、256 fs シリアルクロック周波数を使用し、各フレームの開始時に 16 ビット データワードを送信します。

設定と動作に必要な手順を以下に説明します。

1. コーデックで必要なサンプリング レートとデータワード サイズを決定します。ここではサンプリング レートを 8 kHz と想定します。
2. コーデックで必要なシリアル転送クロック周波数を決定します。ほとんどのコーデックはサンプリング周波数の倍数に当たるシリアルクロック信号を必要とします。この例のコーデックでは 256 fs、または 1.024 MHz の周波数を必要とします。従って、データ転送を開始するには 256 シリアルクロック サイクルごとに 1 つの FS パルスを生成する必要があります。
3. DCI をシリアル転送クロックに対して設定します。
   - CSCK 信号が DCI によって生成される場合、CSCKD 制御ビット (DCICON1<10>) をクリアし、適切なクロック周波数を生成する値を DCICON3 に書き込みます (**20.4.3「ビットクロック ジェネレータ」**参照)。
   - CSCK 信号がコーデックまたはその他の外部ソースによって生成される場合、CSCKD 制御ビットを設定し、DCICON3 レジスタをクリアします。
4. FS 信号をマルチチャンネル モードに設定するには、COFSM 制御ビット (DCICON1<1:0>) をクリアします。
5. DCI が FS 信号を生成する場合 (マスタ)、COFSD 制御ビット (DCICON1<8>) をクリアします。DCI が FS 信号を受信する場合 (スレーブ)、COFSD 制御ビットをセットします。
6. CSCK の立ち下がりエッジで入力データをサンプリングするために、CSCKE 制御ビット (DCICON1<9>) をクリアします。これはほとんどのコーデックで代表的な設定です。正しいサンプリング エッジを確実に使用するには、コーデックのデータシートを参照してください。
7. 所定のデータワード サイズに対応する WS 制御ビット (DCICON2<3:0>) を書き込みます。この例のコーデックでは、16 ビット データワード サイズに対応して WS<3:0> = 1111 と設定する必要があります。
8. フレームごとの所定のデータワード数に対応する COFSG 制御ビット (DCICON2<8:5>) を書き込みます。WS 制御ビットと COFSG 制御ビットがデータフレーム長の CSCK サイクル数を決定します (**20.4.7「フレーム同期ジェネレータ」**参照)。この例のコーデックで要求される 256 ビット データフレームに対応するために、COFSG<3:0> = 1111 と設定します。
9. CSDOM 制御ビット (DCICON1<6>) を使用して CSDO ピンの出力モードを設定します。DCI に接続されたデバイスが 1 つの場合、CSDOM はクリアできます。こうすると、未使用データ タイムスロットの間 CSDO ピンは強制的に「0」になります。CSDO ピンに接続されたデバイスが複数の場合、CSDOM の設定が必要となる場合があります。
10. フレーム内のどのデータ タイムスロットを送受信するかを特定するために、TSCON レジスタと RSCON レジスタにそれぞれ書き込みます。このシングル チャンネルのコーデックでは、TSCON = RSCON = 0x0001 としてデータフレームの最初の 16 ビット タイムスロットでの送受信を有効にします。
11. 所定のデータワード数をバッファするために、BLEN 制御ビット (DCICON2<11:10>) を設定します。このシングル チャンネルのコーデックでは、BLEN = 00 により、データフレームごとに割り込みが発生します。BLEN の値を大きくすると、このコーデックは割り込みの間に複数のサンプルをバッファします。
12. 割り込みを使用する場合、DCIIF ステータスビット (IFS2<9>) をクリアし、DCIIE 制御ビット (IEC2<9>) をセットします。
13. **20.5.1.1「DCI 起動とデータ バッファリング」**に示すように動作を開始します。

20.5.4.2 DMA を使用するマルチチャンネルの設定

DMA を使用するマルチチャンネル DCI の設定は、**20.5.4.1「マルチチャンネルのセットアップの詳細」**で説明した設定したと似ていますが、以下の点が異なります。

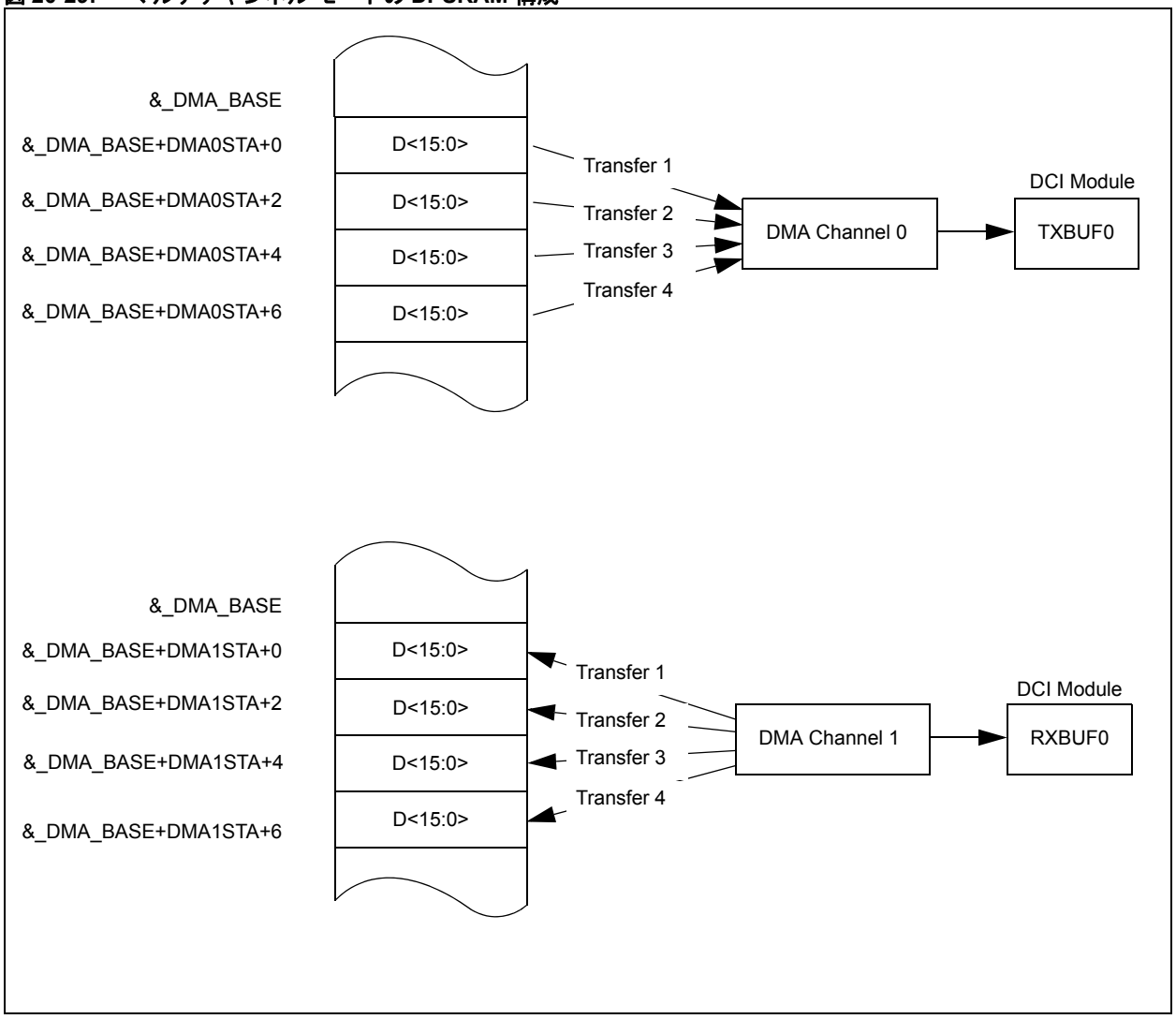
- BLEN は必ず「0」に設定します。それ以外の値に設定すると、予期せぬ結果が生じる可能性があります。
- DMA チャンネルは、必ず RXBUF0/TXBUF0 レジスタに読み書きするように設定します。DMA モジュール設定の詳細は、『dsPIC33F ファミリ リファレンス マニュアル』の**セクション 22.「ダイレクト メモリ アクセス (DMA)」**(DS70182) を参照してください。

DMA 使用手順 :

- 1 つの DMA チャンネルは、DPSRAM (Dual Port SRAM) から読み出し、TXBUF0 レジスタに書き込むように設定します。
- もう 1 つの DMA チャンネルは、RXBUF0 レジスタから読み出し、DPSRAM に書き込むように設定します。
- DMA チャンネルは DCI モジュールを有効化する前に有効にしておく必要があります。これにより、最初の DCI 割り込みが DMA モジュールによって確実に処理されます。

図 20-29 に、この例の DPSRAM 構成を示します。DCI モジュールで有効なタイムスロットは 1 つのみであるため、送信する 16 ビットのデータは、DPSRAM の連続する位置に格納する必要があります。受信するデータは DPSRAM の連続する位置に格納されます。

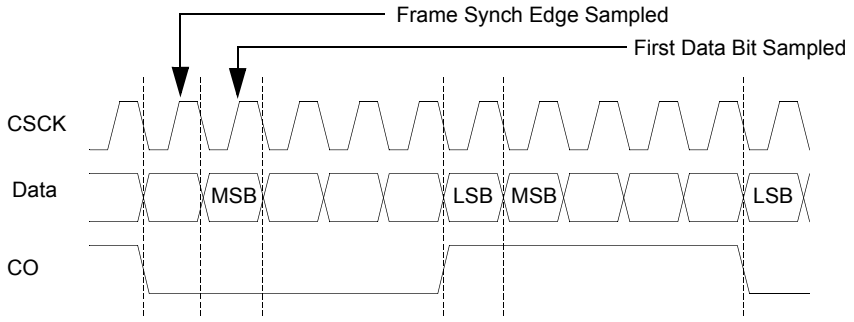
図 20-29: マルチチャンネル モードの DPSRAM 構成



20.5.5 I²S の動作

I²S 動作モードは、デューティ サイクルが 50% の FS 信号を要求するコーデックに使用します。シリアルクロック サイクルの I²S FS 信号の周期は、DCI モジュールに接続されたコーデックのワードサイズによって決まります。図 20-30 に、COFS ピン上の High から Low または Low から High への遷移エッジで区切られる新たなワード境界の始まりを示します。I²S コーデックは一般にステレオまたは 2 チャンネル デバイスで、FS 信号の Low 時間で 1 つのデータワードが転送され、High 時間でもう 1 つのデータワードが送信されます。

図 20-30: I²S インターフェイス フレーム同期タイミング



Note: ここでは、分かりやすく説明するために 5 ビット転送を図示しています。I²S プロトコルはワード長を規定していません。これはシステムによって決まります。

DCI モジュールを I²S モードに設定するには、01h の値を DCICON1 SFR の COFSM<1:0> 制御ビットに書き込みます。I²S モードで動作する場合、DCI モジュールは 50% デューティ サイクルで FS 信号を生成します。FS 信号の各エッジが新たなデータワード転送の境界を示します。ユーザ ソフトウェアは、DCICON2 の COFSG 制御ビットと WS 制御ビットを使用してフレーム長とデータワード サイズも選択する必要があります。

20.5.5.1 I²S の詳細設定

本セクションでは、I²S コーデック向けに DCI を設定する際の手順を説明します。ここでは、仮想の I²S コーデックを使用します。

この例で使用する I²S コーデックでは、64 fs シリアルクロック周波数を使用し、データフレームで 2 つの 16 ビット データワードを送信します。従って、COFS 信号は 32 サイクルの High と 32 サイクルの Low で、フレーム長は 64 CSCK サイクルです。最初のデータワードは COFS の立ち下がりエッジの 1 CSCK サイクル後に送信され、2 番目のデータワードは COFS の立ち上がりエッジの 1 CSCK サイクル後に送信されます。

1. コーデックで使用するサンプリング レートを決定し、CSCK 周波数を決定します。ここでは、fs は 48 kHz と想定します。
2. コーデックで必要とするシリアル転送クロック周波数を決定します。サンプルのコーデックでは 64 fs、または 3.072 MHz の周波数を必要とします。
3. DCI をシリアル転送クロックに対して設定する必要があります。CSCK 信号が DCI によって生成される場合、CSCKD 制御ビット (DCICON1<10>) をクリアし、適切なクロック周波数を生成する値を DCICON3 に書き込みます (**20.4.3「ビットクロック ジェネレータ」**参照)。CSCK 信号がコーデックまたはその他の外部ソースによって生成される場合、CSCKD 制御ビットをセットし、DCICON3 レジスタをクリアします。
4. COFSM<1:0> = 01 にセットし、FS 信号を I²S モードに設定します。
5. DCI が FS 信号を生成する場合 (マスタ)、COFSD 制御ビット (DCICON1<8>) をクリアします。DCI が FS 信号を受信する場合 (スレーブ)、COFSD 制御ビットをセットします。
6. CSCK の立ち上がりエッジで入力データをサンプリングするために、CSCKE 制御ビット (DCICON1<9>) を設定します。これはほとんどの I²S コーデックで代表的な設定です。
7. 所定のデータワード サイズに対応する WS 制御ビット (DCICON2<3:0>) を書き込みます。この例のコーデックでは、16 ビット データワード サイズに対応して WS<3:0> = 1111 と設定します。

8. フレームごとの所定のデータワード数に対応する COFSG 制御ビット (DCICON2<8:5) を書き込みます。WS 制御ビットと COFSG 制御ビットがデータフレーム長の CSCK サイクル数を決定します (**20.4.7「フレーム同期ジェネレータ」**参照)。この例のコーデックでは、COFSG<3:0> = 0001 に設定します。

Note:	I^2S モードでは、COFSG ビットをデータフレームの半分の長さに設定します。サンプルのコーデックでは、COFSG<3:0> = 0001 (フレームごとに 2 データワード) に設定し、32 ビットフレームを生成します。この設定により、64 ビット長の I^2S データフレームが生成されます。

9. CSDOM 制御ビット (DCICON1<6>) を使用して CSDO ピンの出力モードを設定します。DCI に接続されたデバイスが 1 つの場合、CSDOM はクリアできます。CSDO ピンに接続されたデバイスが複数の場合、CSDOM のセットが必要な場合があります。
10. フレーム内のどのデータ タイムスロットを送受信するかを特定するために、TSCON レジスタと RSCON レジスタにそれぞれ書き込みます。このコーデックでは、TSCON = 0x0001、RSCON = 0x0001 に設定して、32 ビット データフレームの最初の 16 ビット タイムスロットでの送受信を有効にします。隣接するタイムスロットを有効にして、16 ビットを超える長さのデータワードをバッファする事が可能です。
11. 所定のデータワード数をバッファするために、BLEN 制御ビット (DCICON2<11:10>) を設定します。2 チャンネルの I^2S コーデックの場合、BLEN<1:0> = 01 により、2 データワード転送後に割り込みが発生します。
12. 割り込みを使用する場合、DCIIF ステータスビット (IFS2<9>) をクリアし、DCIIE 制御ビット (IEC2<9>) を設定します。
13. **20.5.1.1「DCI 起動とデータ バッファリング」**に示すように動作を開始します。I^2S マスタモードでは、モジュールの有効化後に COFS ピンが High に駆動され、TXBUF0 に読み込まれたデータの転送を開始します。

20.5.5.2 DMA を使用する I^2S の設定

DMA を使用する I^2S DCI のセットアップは、**20.5.5.1「I2S の詳細設定」**で説明した設定方法に似ていますが、以下の点が異なります。

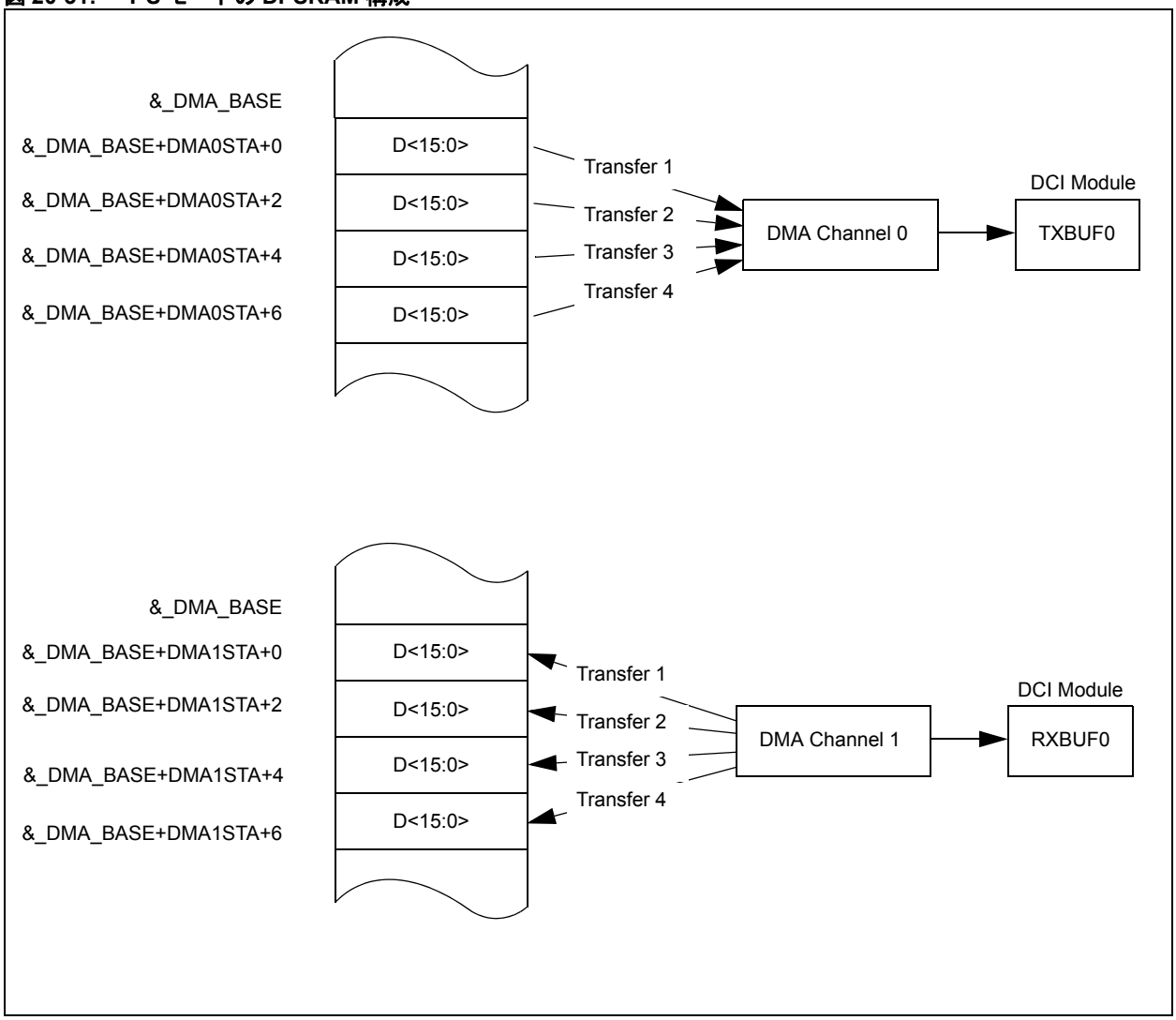
- BLEN は必ず「0」に設定します。BLEN をこれ以外の値に設定すると、予期せぬ結果が生じる可能性があります。
- DMA チャンネルは、必ず RXBUF0/TXBUF0 レジスタに読み書きするように設定します。DMA モジュール設定の詳細は、『dsPIC33F ファミリ リファレンス マニュアル』の**セクション 22.「ダイレクト メモリ アクセス (DMA)」**(DS70182) を参照してください。

DMA 使用手順 :

- 1 つの DMA チャンネルは、DPSRAM (Dual Port SRAM) から読み出し、TXBUF0 レジスタに書き込むように設定します。
- もう 1 つの DMA チャンネルは、RXBUF0 レジスタから読み出し、DPSRAM に書き込むように設定します。
- DMA チャンネルは DCI モジュールを有効化する前に有効にしておく必要があります。これにより、最初の DCI 割り込みが DMA モジュールによって確実に処理されます。

図 20-31 に、この例の DPSRAM 構成を示します。送信データは最初のデータワードとして (COFS 信号の立ち下がりエッジで送信)、続けて 2 番目のデータワードとして構成されます (COFS 信号の立ち上がりエッジで送信)。

図 20-31: I²S モードの DPSRAM 構成



20.5.5.3 I²S チャンネル配置の決定方法

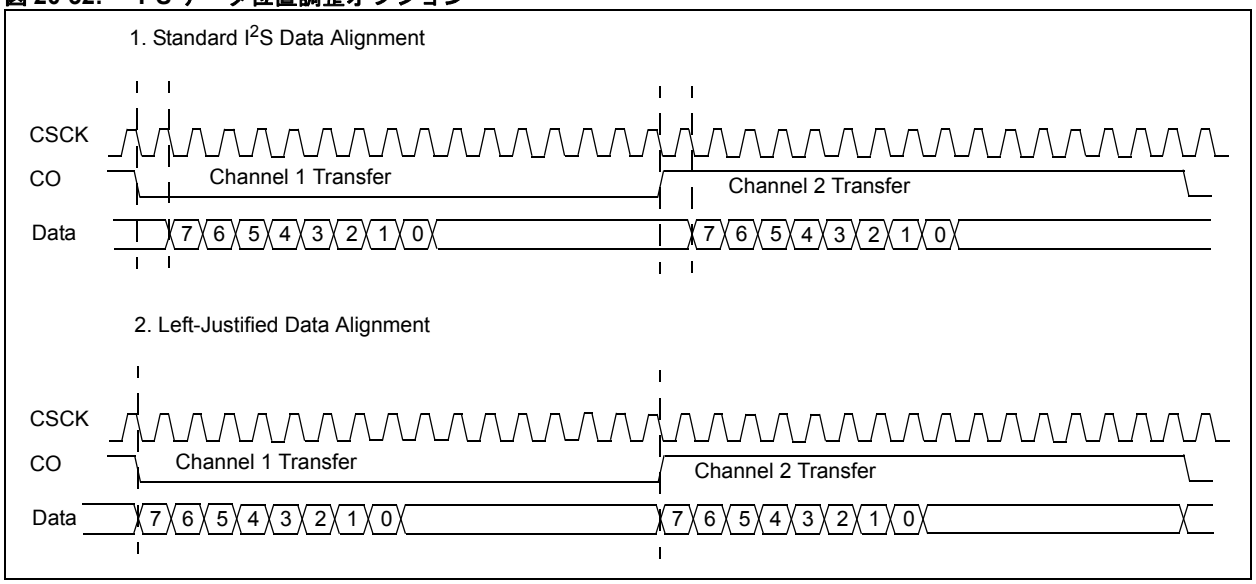
ほとんどの I²S コーデックは 2 つのデータ チャンネルをサポートしており、FS 信号のレベルはデータフレームのどちらか半分で転送するチャンネルを示します。DCI 割り込みサービスルーチンでピンの現在のレベルを判別するために、関連する PORT レジスタを使用して COFS ピンをソフトウェアでポーリングできます。これにより、受信レジスタにどのデータがあるか、次のフレーム転送に備えどのデータを送信レジスタに書き込むべきかを示します。

20.5.5.4 I²S データ位置調整

I²S 仕様通り、既定値でのデータワード転送は FS 信号の遷移の 1 シリアルクロック サイクル後に開始します。DJST 制御ビット (DCICON1<5>) を使用して、「MSb 左寄せ」オプションを選択できます。

DJST = 1 の場合、I²S データ転送は MSb 左寄せです。データワードの MSb は、FS 信号の立ち上がりまたは立ち下がりエッジと同じシリアルクロック サイクルで CSDO ピンに出力されます。データワードの転送終了後、CSDO ピンの状態は CSDOM (DCICON1<6>) ビットによって決まります。

図 20-32: I²S データ位置調整オプション



20.5.6 AC-LINK の動作

本セクションでは、AC-LINK モードでの DCI 使用方法を説明します。AC-LINK モードは、AC'97 準拠のコーデック デバイスと通信します。

20.5.6.1 AC-LINK データフレーム

AC-LINK データフレームは 256 ビットで、16 ビットの制御スロットと、それに続く 20 ビットのデータスロット 12 個に細分されます。AC'97 コーデックは通常、図 20-33 に示されるように、水晶振動子によるシリアル転送クロックを備えています。

コントローラはシリアルクロックを受信し、FS 信号を生成します。データフレーム レートの既定値は 48 kHz です。AC-LINK システムで使用する FS 信号は、データフレームの開始から 16 CSCK 周期の間は High、240 CSCK 周期の間は Low です。

図 20-35 に、FS 信号の立ち上がりエッジの 1 CSCK 周期後にデータ転送が開始される様子を示します。データは受信デバイス側で、CSCK の立ち下がりエッジでサンプリングされます。図 20-34 に示すように、AC-LINK 制御とデータタイムスロットは、使い方がプロトコルで定義されています。

図 20-33: AC-Link 信号接続

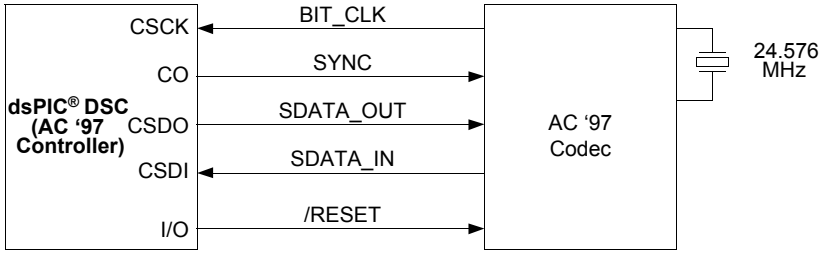


図 20-34:　AC-Link データフレーム

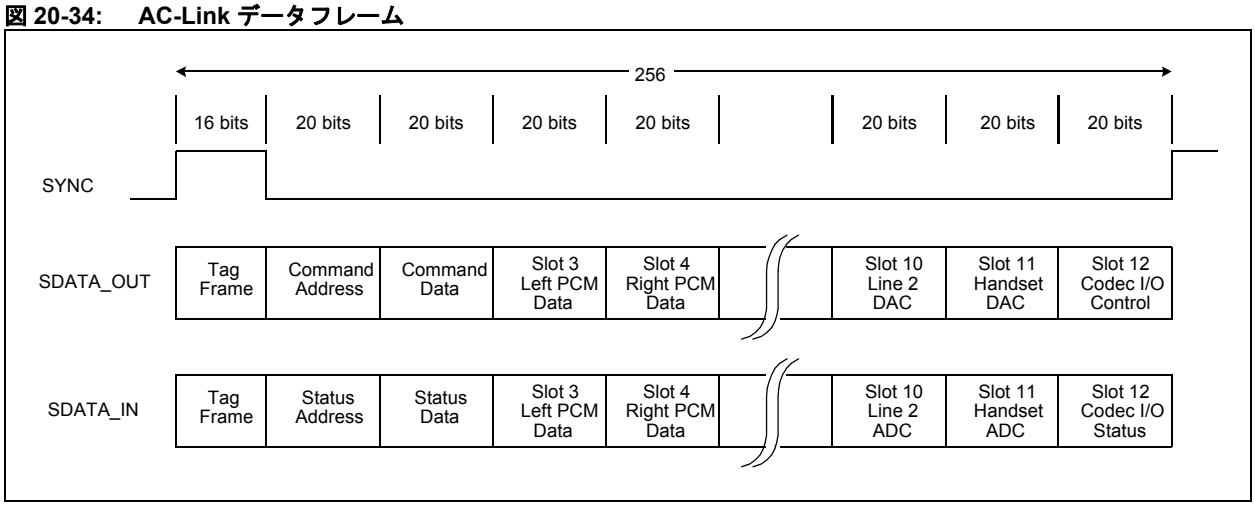
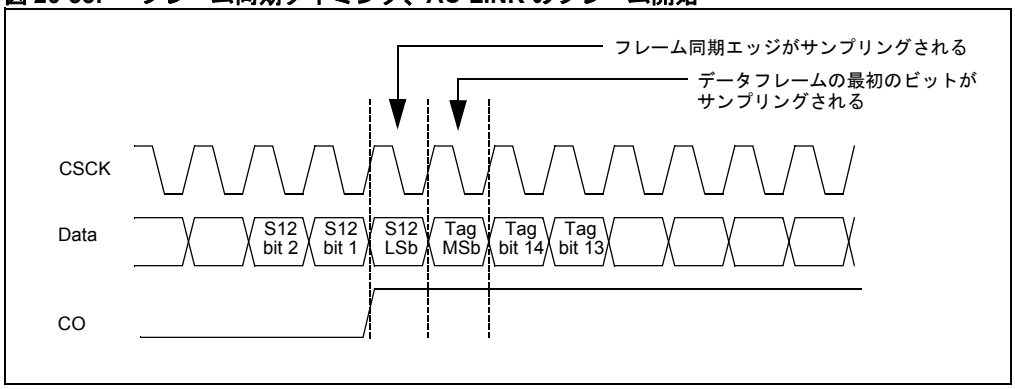


図 20-35:　フレーム同期タイミング、AC-LINK のフレーム開始



DCI モジュールには、AC-LINK プロトコルで定義された、20 ビット データ タイムスロットを活用する 2 つの動作モードがあります。これらの動作モードは、COFSM 制御ビット (DCICON1<1:0>) を使用して選択します。

- 最初の AC-LINK モード (「16 ビット AC-LINK モード」と呼ばれる) を選択するには、COFSM<1:0> = 10 に設定します。
- 2 番目の AC-LINK モード (「20 ビット AC-LINK モード」と呼ばれる) を選択するには、COFSM<1:0> = 11 に設定します。

20.5.6.2　16 ビット AC-LINK モード

16 ビット AC-LINK モードでは、送受信するデータワード長は DCI 送受信レジスタに適合するよう 16 ビットに制限されます。この制限によって影響を受けるのは、AC-LINK プロトコルの 20 ビット データ タイムスロットだけです。受信タイムスロットの場合、入力データは 16 ビットで切り捨てられます。送信タイムスロットの場合、モジュールはデータワードの 4 LSb を「0」に設定します。この動作モードでは、全てのタイムスロットを 16 ビット タイムスロットとして処理する事によって、AC-LINK データフレームを簡略化します。FSG はタイムスロット境界に合わせて維持されます。

20.5.6.3　20 ビット AC-LINK モード

20 ビット AC-LINK モードでは、データ タイムスロット内の全てのビットの送受信が可能ですが、AC-LINK プロトコルで定義される特定のタイムスロット境界に合わせたデータ配置は維持されません。

20 ビット AC-LINK モードは、DCI モジュールのマルチチャンネル モードに似た動作をしますが、生成される FS 信号のデューティ サイクルは異なります。AC-LINK の FS 信号は 16 クロックサイクルの間は High、続く 240 クロックサイクルの間は Low を維持する必要があります。

20 ビットモードは、各 256 ビット AC-LINK フレームを 16 個の 16 ビット タイムスロットとして扱います。20 ビット AC-LINK モードでは、モジュールは COFSG<3:0> = 1111、WS<3:0> = 1111 のように動作します。20 ビット データスロットのデータ配置は、この動作モードでは維持されません。

例えば、TSCON レジスタと RSCON レジスタの全てのビットを設定する事によって、256 ビット AC-LINK データフレーム全体をまとめて送受信する事ができます。利用できるバッファ長の合計は 64 ビットである事から、この AC-LINK フレームを転送するには連続 4 回の割り込みが必要です。アプリケーション ソフトウェアは、SLOT ステータスビット (DCISTAT<11:8>) を監視して現在の AC-LINK フレーム セグメントを追跡する必要があります。

20.5.6.4 AC-LINK の詳細設定

AC-LINK モードを有効にするには、DCICON1 SFR の COFSM<1:0> 制御ビットに 10h または 11h を書き込みます。フレームサイズとワードサイズはプロトコルで設定される事から、ワードサイズ選択ビット (WS<3:0>) と FSG ビット (COFSG<3:0>) は、16 ビット /20 ビット AC-LINK モードには影響を及ぼしません。

ほとんどの AC'97 コーデックは、データ転送を制御するクロック信号を生成します。従って、CSCKD 制御ビットはソフトウェアで設定します。DCI は入力クロック信号から FS 信号を生成するため、COFSD 制御ビットはクリアされます。データを立ち上がりエッジでサンプリングするため、CSCKE 制御ビットはクリアされます。

AC-LINK データフレームでどちらのタイムスロットをバッファするかを選択する必要があり、それに応じてソフトウェアで TSE 制御ビットと RSE 制御ビットを設定します。少なくとも、送受信タグスロットをバッファする必要があります。従って、TSCON<0> 制御ビットと RSCON<1> 制御ビットをソフトウェアで設定します。

Note:	AC-LINK フレームには 13 のタイムスロットがある事から、TSCON<12:0> 制御ビットと RSCON<12:0> 制御ビットのみが、16 ビット AC-LINK モードに影響を及ぼします。

本モジュールを AC-LINK モードに設定する方法は以下の通りです。

1. AC'97 コーデックからシリアル転送クロックを受信するために DCI を設定します。CSCKD 制御ビットをセットし、DCICON3 レジスタをクリアします。
2. 所定の AC-LINK フレーム同期モードを設定するために、COFSM<1:0> 制御ビット (DCICON1<1:0>) を「10b」または「11b」にセットします。
3. COFSD 制御ビット (DCICON1<8>) をクリアすると、DCI は FS 信号を出力します。
4. CSCK の立ち下がりエッジで入力データをサンプリングするために、CSCKE 制御ビット (DCICON1<9>) をクリアします。

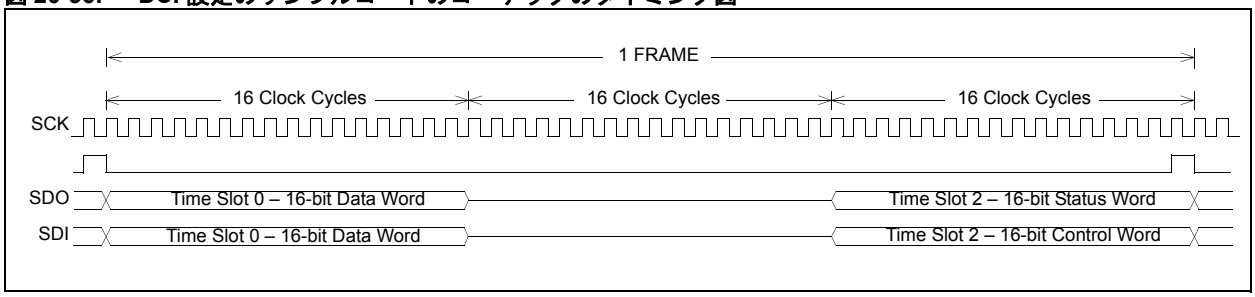
Note:	フレームサイズとワードサイズはプロトコルで設定されるため、ワードサイズ選択ビット (WS<3:0>) と FSG ビット (COFSG<3:0>) は、16 ビット /20 ビット AC-LINK モードには影響を及ぼしません。

5. CSDOM 制御ビット (DCICON1<6>) をクリアします。
6. フレーム内のどのデータ タイムスロットを送受信するかを特定するために、TSCON レジスタと RSCON レジスタにそれぞれ書き込みます。これは、AC-LINK プロトコルのどのデータ タイムスロットを使用するかによって異なります。少なくとも、スロット 0 (タグスロット) での通信が必要です。詳細は、本書の **20.5.6.2「16 ビット AC-LINK モード」**、**20.5.6.3「20 ビット AC-LINK モード」**、補遺を参照してください。
7. 所定のデータワード数をバッファするために、BLEN 制御ビット (DCICON2<11:10>) を設定します。このシングル チャンネルのコーデックでは、BLEN = 00 により、データフレームごとに割り込みが発生します。BLEN の値を大きくすると、このコーデックは割り込みの間に複数のサンプリング結果をバッファします。
8. 割り込みを使用する場合、DCIIF ステータスビット (IFS2<9>) をクリアし、DCIIE 制御ビット (IEC2<9>) をセットします。
9. **20.5.1.1「DCI 起動とデータ バッファリング」**に示すように動作を開始します。

20.6 DCI 設定のサンプルコード

このセクションでは、16 ビットのコーデックを使用してスレーブモードで動作する DCI モジュールの設定について説明します。図 20-36 に、コーデックのタイミング図を示します。コーデックの制御レジスタにアクセスするには、データワードを受信した 16 クロックサイクル後に制御ワードを入力します。コーデックも、データワードを送信した 16 クロックサイクル後にステータスワードを出力します。コーデックはマスタとして設定されており、DCI モジュールの COFS ピンと CSCK ピンを駆動します。

図 20-36: DCI 設定のサンプルコードのコーデックのタイミング図



コーデックが送受信するフレーム内のタイムスロット数は同じである (TSCON = RSCON) ため、ユーザ アプリケーションは連続する TXBUF レジスタに書き込んでデータを送信し、連続する RXBUF レジスタから読み出す事ができます。

例 20-1 に、このコーデックとのインターフェイスのために DCI モジュールを設定するサンプルコードを示します。このサンプルコードでは、2 ワード間隔 (BLEN = 1) で割り込むように DCI モジュールを設定します。この場合、アプリケーションは割り込みごとに 2 つのバッファに書き込むか、2 つのバッファから読み出す必要があります。

または、モジュールが 1 ワード間隔 (BLEN = 0) で割り込むように設定する事もできます。この場合、フレームごとに 2 回の割り込みが生じ、ユーザ アプリケーションは割り込みごとに 1 つのバッファのみを処理する事が求められます。

例 20-1:　DCI 設定のサンプルコード

```
#include 苑 33fxxxx.hî

/* Device configuration registers */
_FGS(GWRP_OFF & GCP_OFF);
_FOSCSEL(FNOSC_PRIPLL);
_FOSC(FCKSM_CSDCMD & OSCIOFNC_OFF & POSCMD_XT);
_FWDT(FWDTEN_OFF);

int main(void)
{
    RSCONbits.RSE2=1;      /* Enable Receive Time Slot 2 */
    RSCONbits.RSE0=1;      /* Enable Receive Time Slot 0 */


    TSCONbits.TSE2=1;      /* Enable Transmit Time Slot 2 */
    TSCONbits.TSE0=1;      /* Enable Transmit Time Slot 0 */

    DCICON1bits.COFSM = 0; /* Multichannel Frame Sync mode */
    DCICON1bits.DJST = 0;  /* Data TX/RX is begun one serial clock cycle after frame sync pulse */
    DCICON1bits.CSCKE = 0; /* Data changes on rising edge sampled on falling edge of CSCK */
    DCICON1bits.COFSD = 1; /* Frame sync driven by codec */
    DCICON1bits.CSCKD = 1; /* Clock is input to DCI from codec */

    DCICON2bits.BLEN = 1;  /* Two data words will be buffered between interrupts */
    DCICON2bits.COFSG = 2; /* Data frame has 3 words */
    DCICON2bits.WS = 15;   /* Data word size is 16 bits*/

    DCICON3 = 0;           /* BCG value is zero since clock is driven by codec */


    IPC15bits.DCIIP=6;     /* Enable the interrupts */
    IFS3bits.DCIIF=0;
    IEC3bits.DCIIE=0;

    TXBUF0= 0x0001;        /* This is the data word */
    TXBUF1= 0x0002;        /* This is the control word */

    DCICON1bits.DCIEN = 1;/* Enable the module */

    while(1);
}

void __attribute__((__interrupt__, no_auto_psv)) _DCIInterrupt(void)
{
    int dataWord;
    int statusWord;

    IFS3bits.DCIIF = 0;
    TXBUF0 = 0x0001;       /* Write some data */
    TXBUF1 = 0x0002;       /* This is the control word */

    dataWord = RXBUF0;     /* Read the data word */
    statusWord = RXBUF1;   /* Read the status word */
}
```

20.7 DMA を使用する DCI モジュール バッファへのデータ転送

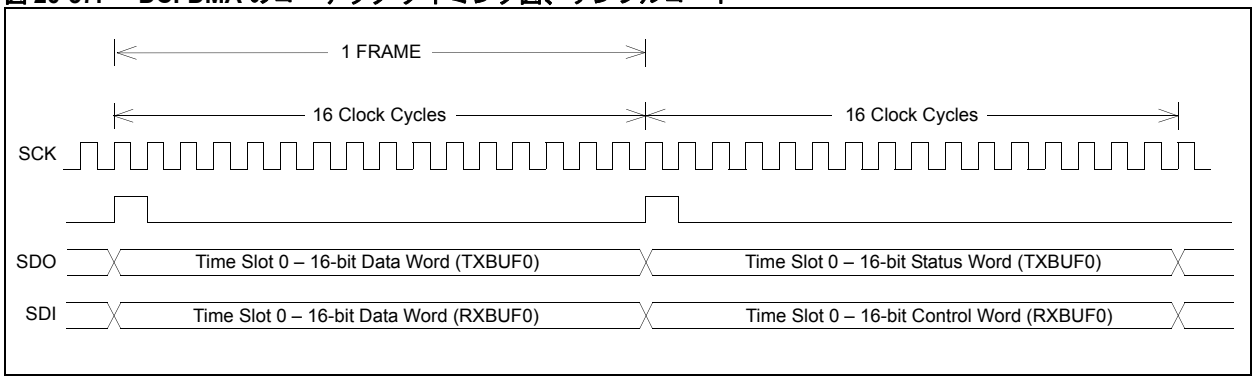
dsPIC33F のダイレクト メモリ アクセス (DMA) モジュールを使用すると、ユーザ アプリケーションが介入しなくても、DPSRAM から DCI モジュール バッファにデータを転送できます。これには少なくとも 2 つの DMA チャンネルが必要です。1 つの DMA チャンネルが受信レジスタからデータを読み出し、他方のチャンネルが送信レジスタにデータを書き込みます。どちらの DMA チャンネルも DCI 転送完了割り込みを使用します。

DCI RXBUF/TXBUF レジスタは 16 ビット レジスタである事から、DMA チャンネルはワード転送に設定する必要があります。バイト値を DCI モジュールに書き込むために、ユーザ ソフトウェアは最初にワードの上位バイト方向に左シフトする必要があります。

例 20-2 に、DMA を連続ピンポンバッファ モードに設定するコードを示します。図 20-37 に、DCI コーデック通信のタイミング図を示します。ピンポンバッファ モードでは、DMA モジュールはデータフレームが格納されるメモリ位置を交互に切り換えます。この方法により、処理済みのデータフレームを送信し、新たなデータフレームを受信している間に、1 つのデータフレームを容易に処理できます。DCI モジュールは、転送完了割り込みごとに DMA モジュールに転送を要求します。

DMA モジュールの詳細は、**セクション 22「ダイレクト メモリアクセス (DMA)」** (DS70182) を参照してください。

図 20-37: DCI-DMA のコーデック タイミング図、サンプルコード



例 20-2: DMA を使用する DCI モジュール バッファへのデータ転送、サンプルコード

```
#include 苑 33fxxxx.hî

/* Device configuration registers */
_FGS(GWRP_OFF & GCP_OFF);
_FOSCSEL(FNOSC_PRIPLL);
_FOSC(FCKSM_CSDCMD & OSCIOFNC_OFF & POSCMD_XT);
_FWDT(FWDTEN_OFF);

#define FCY 40000000
#define CODEC_SAMPLE_RATE 8000
#define DCI_BCG_VALUE( ( (FCY/32) / CODEC_SAMPLE_RATE ) - 1 )
#define FRAME 80

int txBufferA[FRAME] __attribute__((space(dma)));
int txBufferB[FRAME] __attribute__((space(dma)));
int rxBufferA[FRAME] __attribute__((space(dma)));
int rxBufferB[FRAME] __attribute__((space(dma)));

volatile int rxBufferIndicator = 0;

void DCIInit(void);
void processRxData(int * sourceBuffer, int * targetBuffer);
void DMAInit(void);

int main(void)
{
    CLKDIV = 0;               /* Set up for 40 MIPS */
    PLLFBD = 30;
    while (!OSCCONbits.LOCK);

    DMAInit();
    DCIInit();

    while(1);

}

void DCIInit(void)
{
    TSCON = 0x0001;           /* Only one transmit time slot */
    RSCON = 0x0001;           /* Only one receive time slot */

    DCICON1 = 0;
    DCICON1bits.DCIEN = 1;    /* Module is enabled */
    DCICON1bits.DCISIDL = 0;  /* Continue operation in idle */
    DCICON1bits.DLOOP = 0;    /* Loopback mode is disabled */
    DCICON1bits.CSCKD = 0;    /* DCI is master - CSCK is output */
    DCICON1bits.CSCKE = 0;    /* Data is sampled on falling edge */
    DCICON1bits.COFSD = 0;    /* DCI is master - COFS is output */
    DCICON1bits.UNFM = 0;     /* Transmit zeroes on TX underflow */
    DCICON1bits.CSDOM = 0;    /* Transmit 0 on disabled time slots */
    DCICON1bits.DJST = 1;     /* COFS and CSDO start together */
    DCICON1bits.COFSM = 0;    /* DCI mode is multi-channel FS mode */

    DCICON2 = 0;
    DCICON2bits.BLEN = 0;     /* Interrupt on one buffer */
    DCICON2bits.COFSG = 0;    /* Data frame has one word */
    DCICON2bits.WS = 0xF;     /* Word size is 16 bits */

    DCICON3 = DCI_BCG_VALUE;

    _DCIIE = 0;               /* Disabled since DMA is used */

}
void disableInterrupts(void)
{
    /* DMA 0 - DPSRAM to DCI */
```

例 20-2: DMA を使用する DCI モジュール バッファへのデータ転送、サンプルコード (続き)

```
    DMA0CONbits.SIZE = 0;      /* Word transfers */
    DMA0CONbits.DIR = 1;       /* From DPSRAM to DCI */
    DMA0CONbits.AMODE = 0;     /* Register Indirect with post-increment mode */
    DMA0CONbits.MODE = 2;      /* Continuous ping pong mode enabled */
    DMA0CONbits.HALF = 0;      /* Interrupt when all the data has been moved */
    DMA0CONbits.NULLW = 0;
    DMA0REQbits.FORCE = 0;     /* Automatic transfer */
    DMA0REQbits.IRQSEL = 0x3C;/* Codec transfer done */

    DMA0STA =__builtin_dmaoffset(txBufferA);
    DMA0STB =__builtin_dmaoffset(txBufferB);

    DMA0PAD = (int)&TXBUF0;
    DMA0CNT = FRAME-1;

    /* DMA 2 - DCI to DPSRAM */

    DMA2CONbits.SIZE = 0;      /* Word transfers */
    DMA2CONbits.DIR = 0;       /* From DCI to DPSRAM */
    DMA2CONbits.HALF = 0;      /* Interrupt when all the data has been moved */
    DMA2CONbits.NULLW = 0;     /* No NULL writes - Normal Operation */
    DMA2CONbits.AMODE = 0;     /* Register Indirect with post-increment mode */
    DMA2CONbits.MODE = 2;      /* Continuous mode ping pong mode enabled */

    DMA2REQbits.FORCE = 0;     /* Automatic transfer */
    DMA2REQbits.IRQSEL = 0x3C;/* Codec transfer done */

    DMA2STA =__builtin_dmaoffset(rxBufferA);
    DMA2STB =__builtin_dmaoffset(rxBufferB);

    DMA2PAD = (int)&RXBUF0;
    DMA2CNT = FRAME-1;

    _DMA2IP = 5;
    _DMA2IE = 1;

    DMA0CONbits.CHEN = 1;      /* Enable the channel */
    DMA2CONbits.CHEN = 1;

}

void processRxData(int * sourceBuffer, int * targetBuffer)
{
    /* This procedure loops back the received data to the */
    /* the codec output. The user application could process */
    /* this data as per application requirements. */

    int index;

    for(index = 0;index < FRAME;index ++)
    {
        targetBuffer[index] = sourceBuffer[index];
    }

}

void __attribute__((__interrupt__,no_auto_psv)) _DMA2Interrupt(void)
{
    _DMA2IF = 0;               /* Received one frame of data */

    if(rxBufferIndicator == 0)
    {
        processRxData(rxBufferA,txBufferA);
    }
    else
    {
        processRxData(rxBufferB,txBufferB);
    }
    rxBufferIndicator ^= 1;    /* Toggle the indicator */

}
```

20.8 省電力モード時の動作

20.8.1 CPU アイドルモード

DCI モジュールは、CPU のアイドルモード時に動作を継続する事もできます。DCISIDL 制御ビット (DCICON1<13>) によって、CPU のアイドルモード時に DCI モジュールを動作させるかどうかが決まります。

- DCISIDL 制御ビットがクリアされている場合 (既定値)、アイドルモードでもモジュールは通常通り動作を継続します。
- DCISIDL ビットをセットした場合、CPU がアイドルモードへ切り換わるとモジュールは停止します。

20.8.2 スリープモード

CSCK 信号がデバイス命令クロック TCY をクロック源としている場合、DCI はデバイスがスリープモードの間は動作しません。

ただし、DCI モジュールはスリープモードの間も動作が可能で、CSCK 信号が外部デバイスから供給される場合 (CSCKD = 1)、CPU をウェイクアップします。DCI 割り込みイネーブルビット DCIIE は、スリープモードからのウェイクアップ イベントが可能になるように設定しておく必要があります。DCI 割り込みフラグ DCIIF を設定すると、デバイスはスリープモードからウェイクアップします。DCI 割り込み優先度が現在の CPU 優先度よりも高い場合、プログラムの実行は DCI ISR から再開します。そうでない場合、スリープモードに移行させた PWRSAV 命令の次の命令から実行を再開します。

20.8.3 Doze モード

DCI モジュールは Doze モードの影響を受けません。ただし、Doze モードの間は DCI 割り込みに応答する十分な時間がプロセッサにない場合があります。

20.9 DCI に関連するレジスタ

表 20-2 に、DCI モジュールに関連するレジスタの一覧を示します。

表 20-2: DCI レジスタマップ

名称	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	リセット時の値
IFS2	—	—	—	FLTBIF	FLTAIF	LVDIF	DCIIF	QEIIF	PWMIF	C2IF	INT4IF	INT3IF	OC8IF	OC7IF	OC6IF	OC5IF	0000 0000 0000 0000
IEC2	—	—	—	FLTBIE	FLTAIE	LVDIE	DCIIE	QEIIE	PWMIE	C2IE	INT4IE	INT3IE	OC8IE	OC7IE	OC6IE	OC5IE	0000 0000 0000 0000
IPC10	—	FLTAIP<2:0>		—	LVDIP<2:0>			—	DCIIP<2:0>			—	QEIIP<2:0>				0100 0100 0100 0100
DCICON1	DCIEN	—	DCISIDL	—	DLOOP	CSCKD	CSCKE	COFSD	UNFM	CSDOM	DJST	—	—	—	COFSM<1:0>		000- -000 000- --00
DCICON2	—	—	—	—	BLEN<1:0>		—	COFSG<3:0>				—	WS<3:0>				---- 00-0 000- 0000
DCICON3	—	—	—	—	BCG<11:0>												---- 0000 0000 0000
DCISTAT	—	—	—	—	SLOT<3:0>				—	—	—	—	ROV	RFUL	TUNF	TMPTY	---- 0000 ---0 0000
TSCON	TSE15	TSE14	TSE13	TSE12	TSE11	TSE10	TSE9	TSE8	TSE7	TSE6	TSE5	TSE4	TSE3	TSE2	TSE1	TSE0	0000 0000 0000 0000
RSCON	RSE15	RSE14	RSE13	RSE12	RSE11	RSE10	RSE9	RSE8	RSE7	RSE6	RSE5	RSE4	RSE3	RSE2	RSE1	RSE0	0000 0000 0000 0000
RXBUF0	受信 0 データレジスタ																uuuu uuuu uuuu uuuu
RXBUF1	受信 1 データレジスタ																uuuu uuuu uuuu uuuu
RXBUF2	受信 2 データレジスタ																uuuu uuuu uuuu uuuu
RXBUF3	受信 3 データレジスタ																uuuu uuuu uuuu uuuu
TXBUF0	送信 0 データレジスタ																0000 0000 0000 0000
TXBUF1	送信 1 データレジスタ																0000 0000 0000 0000
TXBUF2	送信 2 データレジスタ																0000 0000 0000 0000
TXBUF3	送信 3 データレジスタ																0000 0000 0000 0000

凡例 : r = 予約、x = 未知、u = 不変 網掛けした部分は、将来のモジュール拡張に備えて SFR マップで予約されている位置を表します。予約されている位置は「0」として読み出されます。

Note: SFR アドレスは dsPIC33F のデバイスによって異なります。詳細は、各デバイスのデータシートを参照してください。

20.10 設計のヒント

質問 1: **DCI は 16 ビットを超えるデータワード長をサポートできますか。**

回答 : はい、できます。複数の送受信レジスタを使用してロングワード データを送受信できます。詳細は、**20.5.3「ロングワード データ サポートのデータ パッキング」**を参照してください。

質問 2: **dsPIC33F はコーデック クロック源を使用できますか。**

回答 : はい、できます。コーデック クロックは、dsPIC33F の適切なクロック入力ピンに接続する必要があります。これにより、dsPIC33F とコーデックを同期できます。

質問 3: **HD オーディオとは何ですか。DCI モジュールでサポートされていますか。**

回答 : Intel® 高解像度 (HD) オーディオは、コーデック インターフェイス向けの新しいプロトコルです。DCI モジュールは、このプロトコルをサポートしていません。

20.11 関連アプリケーション ノート

本セクションに関連するアプリケーション ノートの一覧を以下に示します。一部のアプリケーション ノートは dsPIC33F デバイス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。DCI モジュールに関連する現在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

現在関連するアプリケーション ノートはありません。

Note: dsPIC33F 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。

20.12 改訂履歴

リビジョン A (2007 年 5 月)

本書の初版

リビジョン B (2008 年 9 月)

改訂内容 :

- 図 :
  - ロングワード データのデータ パッキングの例 (図 20-26 参照)。TXBUF0 の内容はデータ ワードを 23:8 として読み出す
- 上記に加えて、表現および体裁の変更等、本書全体の細部を修正

ISBN: 978-1-60932-869-6